A CONFORMAL MAPPING GRID GENERATION METHOD FOR MODELING HIGH-FIDELITY AEROELASTIC SIMULATIONS

A Thesis

by

GREGORY DORWAY WORLEY

Submitted to the Office of Graduate Studies of
Texas A&M University
in partial fulfillment of the requirements for the degree of

MASTER OF SCIENCE

May 2010

Major Subject: Aerospace Engineering

A CONFORMAL MAPPING GRID GENERATION METHOD FOR MODELING HIGH-FIDELITY AEROELASTIC SIMULATIONS

A Thesis

by

GREGORY DORWAY WORLEY

Submitted to the Office of Graduate Studies of
Texas A&M University
in partial fulfillment of the requirements for the degree of

MASTER OF SCIENCE

Approved by:

Chair of Committee,	Paul Cizmas
Committee Members,	Bojan Popov
	Othon Rediniotis
	Thomas Strganac
Head of Department,	Dimitris Lagoudas

May 2010

Major Subject: Aerospace Engineering

ABSTRACT

A Conformal Mapping Grid Generation Method for Modeling
High-Fidelity Aeroelastic Simulations. (May 2010)
Gregory Dorway Worley, B.S., Texas A&M University
Chair of Advisory Committee: Dr. Paul Cizmas

This work presents a method for building a three-dimensional mesh from two-dimensional topologically identical layers, for use in aeroelastic simulations. The method allows modeling of large deformations of the wing in both the span direction and deformations in the cord of the wing. In addition, the method allows for the modeling of wings attached to fuselages. The mesh created is a hybrid mesh, which allows cell clustering in the viscous region. The generated mesh is high quality and allows capturing of nonlinear fluid structure interactions in the form of limit cycle oscillation.

To my wife

ACKNOWLEDGMENTS

I would like to thank my advisor and committee chair, Dr. Cizmas, for his help and guidance during my research and graduate education. I would also like to thank my committee members, Dr. Thomas Strganac, Dr. Othon Redinotis, and Dr. Bojan Popov, for their help and support during my graduate studies.

I would also like thank Dr. Cizmas's other grad students, Brian, Dave, Forrest, Raymond, Robert, Tom, and Will, who helped me with both my graduate studies and my research.

Finally, I would like to thank my wife for financing my way through college, putting up with my work habits, and supporting me during the course of my research.

TABLE OF CONTENTS

CHAPTER Page

LIST OF TABLES

LIST OF FIGURES

FIGURE Page

NOMENCLATURE

α	-	angle of attack
C_x	-	denotes a coefficient
CFD	-	computational fluid dynamics
FFT	-	Fast Fourier Transform
Γ	-	Circulation
I_{max}	-	number of nodes around the airfoil
J_{max}	-	number of layers in the O-grid
K_{max}	-	number of span layers
LCO	-	limit cycle oscillation
R	-	singularity in the Wieing-Manea equation
RANS	-	Reynold-averaged Navier-Stokes
STL	-	StereoLithography
t	-	solidity
τ	-	ellipse angle
TDT	-	Transonic Dynamics Tunnel
ζ	-	coordinate in $\mathscr{K}$ domain
z_2	-	coordinate in $\mathscr{E}$ domain
z_1	-	coordinate in $\mathscr{C}$ domain
z_p	-	coordinate in $\mathscr{R}$ domain

Subscripts

D	-	total drag
ir	-	span layer number ir
i	-	i^{th} node

j - j^{th} node

L - total lift

∞ - value at the inlet

Superscripts

$*$ - total temperature, or pressure ect.

CHAPTER I

INTRODUCTION

A. Statement of Work

The objective of this work was to develop a mesh generation method that is compatible with an existing aeroelastic solver [2], while generating a high quality mesh around a wing and wing/fuselage combination. Two interrelated mesh generation methods were developed: (1) a more robust and accurate wing mesh generation method based on work done by Kim [3], and (2) a mesh generation method for a wing/fuselage configuration that used the wing mesh as a basis for the wing/fuselage mesh. Both the wing mesh and the wing/fuselage mesh consisted of topologically identical layers. Each layer consisted of a quadrilateral structured O-grid generated using a conformal mapping algorithm surrounded by an unstructured triangular mesh. The wing/fuselage mesh was generated in two steps: (1) generating the surface mesh on the fuselage, and (2) generating the wing/fuselage volume mesh. The surface mesh was generated by projecting the base layer of the wing mesh onto a background fuselage mesh and then refining the mesh to conform to the fuselage. Then the volume mesh was generated using a modified 2.5D meshing method with mesh smoothing. In order to generate a mesh capable of simulating viscous flows, cells were clustered around the fuselage and wing surfaces.

The journal model is *International Journal for Numerical Methods in Fluids.*

B. Motivation

Aeroelasticity is the study of the interaction of three forces: (1) inertial, (2) elastic, and (3) aerodynamic [4]. Aeroelasticity can be broken down into four different disciplines depending on which forces are interacting. A diagram of these forces can be seen in Fig. 1.

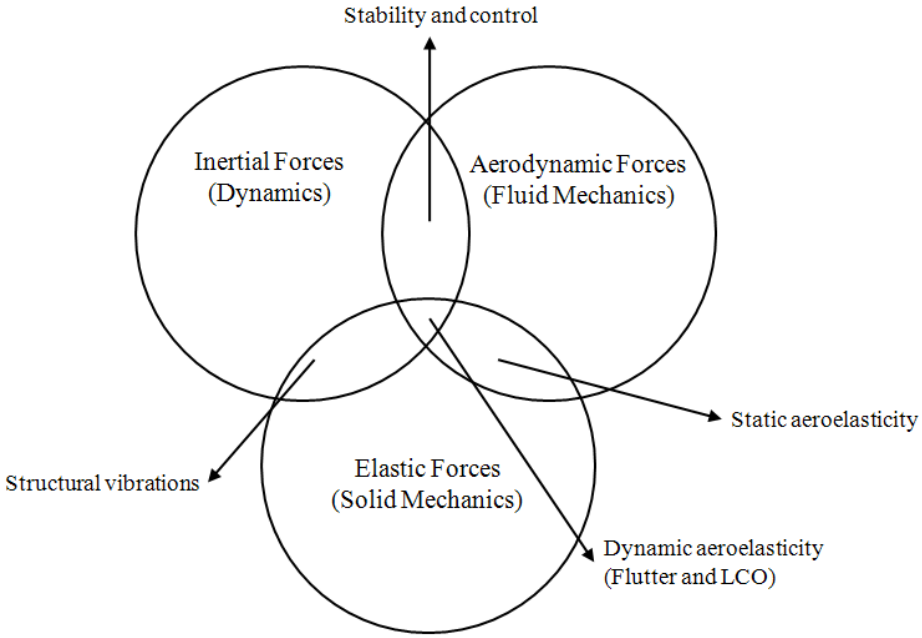


Fig. 1. Relationship between the three forces.

Static aeroelasticity is the interaction between the elastic and aerodynamic forces. This discipline includes some of the more the common aeroelastic phenomena such as divergence, control reversal and lift effectiveness. Divergence occurs when the deformation-dependent aerodynamic loads on the structure are greater than the stiffness-dependent restorative forces and usually results in destruction of the wing. Control reversal is caused by the flexibility of the structure and causes control inputs

to have no effect. As the control surface deflects, the structure reacts in the opposite direction, which results in a net control force of zero. Lift effectiveness is the term used to describe the change in the lift characteristics of a structure due to structural deformations.

The interaction of the aerodynamic and inertial forces is the stability and control discipline, where the structure is assumed to be rigid. The interaction of the inertial and structural forces is the structural dynamics discipline, which deals with structural vibrations.

The final discipline is the dynamic aeroelasticity discipline and it deals with the interaction of all three forces. Two common phenomena associated with this discipline are flutter and Limit Cycle Oscillations (LCO). These phenomena occur when the areodynamic loads excite one or more of the structural modes. Flutter is the unbounded growth in amplitude of wing deformations, which ultimately results in wing failure. The LCO instability is when the wing deformations increase asymptotically to a certain amplitude.

Conventional methods for studying aeroelastic phenomena rely on linear aeroelastic models. While these linear models can predict divergence, flutter and control reversal, they are unable to predict the onset of LCO [5]. This is due to the inability of the linear models to capture the inherent aerodynamic and structural nonlinearities that cause LCO's.

Nonlinear aeroelastic analysis can provide a clearer picture of the aeroelastic phenomena. This requires coupling a full order flow solver with a nonlinear structural solver. With these tools, LCO motion can be described as well as other nonlinear behavior such as internal resonances and chaotic motion. However, fully nonlinear analysis is computationally expensive and requires an adaptable, high quality mesh for the flow solver.

As new classes of aircraft with highly flexible wings are developed, the importance of study into dynamic aeroelastic phenomena associated with highly flexible wings has become increasingly evident. There is evidence of adverse aeroelastic responses due to system nonlinearities in both the new classes of aircraft and older fighter aircraft. LCOs have been observed on F-16 fighters with store configurations and remain a persistent challenge [6, 7]. Such LCOs are unacceptable since aircraft performance, aircraft certification, mission capability, and human factor issues such as pilot fatigue are adversely affected [8].

An important step in fully nonlinear aeroleastic analysis is the generation of a high-fidelity mesh. Due to the highly flexible nature of these new classes of aircraft, the mesh must be able to model large out-of-plane and torsional wing deformations, while maintaining cell quality. To fully explore aeroelastic phenomena, the mesh must also be able to accurately model both the fuselage and the wing.

C. Background

This section presents the background information on mesh generation and deformation techniques typically used in aeroelastic analysis. The method used in this work are also briefly outlined in this section.

Different approaches have been developed to address the deforming mesh problem. One mesh deformation algorithm was developed by Allen [9] and can be used with any mesh type, however, the implementation costs for both the mesh generation algorithm and the flow solver are high. In most cases, the deformation algorithm employed is dependent on the mesh type and the application. For meshes generated by using a multi-block method, such as ZEUS [10] and OVERGRID [11], a natural way to account for the structural deformation is to slide the overlapping regions of

the grids [12, 13] across each other. The main advantage of these methods is that the grids on the surface of the object do not deform as the object moves. However, this approach requires the interpolation algorithm that communicates between the overlapping grids to be updated for each step. In addition, overset grid generation has a high implementation cost.

Another widely used mesh deformation algorithm is the tension spring analogy [14]. In this method, each edge of the mesh is represented by a spring for which the stiffness is proportional to the reciprocal of the length of the edge. Since the mesh is represented by a network of springs, any boundary deformation will translate into a deformation of the spring network, which adjusts the shape of the mesh until equilibrium is reached. The displacements in each direction are decoupled and the equation that updates the position of the nodes is relatively easy to solve. A disadvantage of this method is that for highly distorted meshes, collapsed or negative volume cells may appear, especially in high-aspect ration cells such as those used for viscous flows. Due to its ease of implementation and low computational cost, this method is employed in this work.

One of the more widely used mesh types that utilize the tension spring analogy is a hybrid mesh because they allow better control of the cell size in the viscous region [15] while still allowing the tension spring analogy to deform the mesh in the rest of the domain. A common hybrid mesh combination is prism cells for the viscous region and tetrahedral cells for the rest of the domain. Advancing-layers grid generation technique is a popular way to cluster cells in the viscous region [16], however, this method has high implementation costs.

To develop a simple yet effective volume mesh generation tool for a deformable wing, the present work uses a mapping method and mesh smoothing techniques to generate a hybrid wing mesh. Mapping, also know as 2.5D Meshing, is a popular

volume mesh generation method used extensively in finite element analysis [17, 18], which maps a 2D source mesh in a sweeping motion to generate a volume mesh. The resulting mesh has topologically identical layers. The hybrid mesh addresses the problem of collapsed cells in the viscous region when using a tension spring analogy by generation an O-grid around the wing.

The 2D source mesh can be generated using a variety of methods. Some of the more common types of 2D mesh generation include algebraic grid generation, advancing front, octree, Delaunay triangulations, conformal mapping [19, 20]. Octree is at an immediate disadvantage due to its difficulty in capturing the surface of complex shapes [17]. Delaunay triangulation is attractive due to its ability to generate high quality cells and its efficient use of the computational domain. This method, however, does not allow for direct control over cell clustering at the boundaries and only produces triangular cells [21]. 2D advancing front has the same high implementation cost as its 3D counterpart [22]. Algebraic grid generation has low implementation cost and high computational efficiency, but this method can have difficulty generating high quality cells over the full domain. Conformal mapping, while mathematically complex, allows direct control over cell spacing and allows for generation of quadrilateral cells around the wing surface [23]. However, conformal mapping has difficulty meshing arbitrary domains.

When modeling the wing/fuselage interaction, the grid generation algorithm must be adapted to capture the fuselage curvature. While the 2.5D Meshing does an excellent job at generating a mesh around a wing, there are potential problems while projecting the source mesh onto the fuselage surface. Therefore, the wing mesh generation method was modified and expanded to generate a wing/fuselage mesh.

For a wing/fuselage configuration, the fuselage surface must be discretized. Surface discretization methods can be generally classified as direct or indirect. Direct

methods [24, 25] generate a mesh directly on the surface of the shape, which requires a mathematical description of the 3D surface. Due to the mathematical complexity of describing a 3D shape, direct methods are an attractive option only for surfaces with relatively simple shapes. Indirect methods [26, 27] utilize the bijective mapping between a triangulated planar parametric surface and the desired 3D surface. Since these methods require the mapping to exist, only certain surfaces can be meshed. In addition, problems arise due to distortion and stretching when the parametric surface is mapped. Special care must be taken to deal with singular cases that could arise from these deformations [22].

Two of the most common direct methods are surface refinement [25] and advancing front [28]. Advancing front methods allow for cell clustering on high curvature surfaces, however, they are difficult to implement. The main disadvantage of the surface refinement method is the requirement of a high quality mesh as a starting point. However, if a quality mesh is available, this method is more computationally efficient than the advancing front method and can produce high quality meshes. Work done by Bechet et al. [29] uses stereolithography (STL) files as a starting mesh for mesh refinement. However, STL data often have ill-conditioned or missing faces and can be difficult to work with as a starting mesh [30]. STL data are better suited for use as a background mesh.

To replicate the simplicity and efficacy of the wing mesh generation, the present work uses the wing mesh as the basis for the wing/fuselage mesh. The base layer of the wing mesh is used as the starting mesh for the surface refinement. The background mesh for the fuselage is supplied by the data from an STL file and the fuselage boundary is delineated within the wing mesh using an 1D advancing front method [28, 31]. The topologically identical layers allow the translation of the mesh refinement and deformation of the base layer to the rest of the mesh.

D. Flow Solver and Mesh Deformation Algorithm

The flow solver used to test the meshes in this work was developed by Cizmas and Han [32] and improved by Kim [3]. The flow solver is an unstructured mixed-type grid with a finite-volume discretization. The code is capable of solving both Euler and Navier-Stokes equations. Turbulence is modeled using a two-equation $k-\omega$ SST turbulence model.

The mesh deformation algorithm used in this work was developed by Gargoloff et al. [2]. This method utilized a tension spring analogy algorithm to deform the mesh to allow for wing displacement. During the deformation, the mesh connectivity was not modified.

E. Parallel Algorithm

Parallel processing is a powerful tool for significantly decreasing the computational time for a simulation [2]. In order to run a simulation in parallel, the computational domain must be split into segments. Each segment of the domain is run on a separate processor and information is passed between the sections. Any mesh type can be split into these domains, however, some methods lend themselves more readily to segmentation of the domain. Multi-block meshes are already split into segments and are easy to run in parallel. Meshes that have topologically identical layers also lend themselves well to simple partitioning for parallel processing due to the structured nature in the spanwise direction. Since the topology is identical across the layers, communication for parallel processing is simplified.

The parallization of the flow solver was implemented by Gargoloff [33], by using message-passing interface (MPI) standard libraries for communication between the processors. The computational mesh was split into sub-meshes that included one

or more topologically identical layer. The mesh generation method presented in this work uses topologically identical layers which allows this algorithm to still be utilized.

F. Original Contributions of the Work

- Development of a 2D conformal mapping grid generation method.
- Integration of the 2D conformal mapping grid generator with an existing 3D wing mesh generator.
- Development of a surface mesh refinement method
- Development of a volume meshing algorithm for a wing/fuselage combination

G. Outline of Thesis

This thesis presents a 2.5D mesh generation algorithm that generates a high quality 3D hybrid mesh. Chapter II introduces the method used to generate a 3D mesh around a stand-alone wing. Chapter III presents the method for projecting and refining the 3D wing mesh to fit onto a fuselage. This chapter includes the strategy used for edge switching, node deformation and addition, and node projection. Chapter IV presents the validation of both the wing mesh and the wing/fuselage mesh. This chapter also includes the comparison of results from the original wing mesh generator to results from the updated mesh generator. The test cases used were: turbulent flow around a infinite NACA 0012, transonic flow around a finite M6 ONERA wing, transonic flow around the DLR-F6 wing/fuselage configuration and the aeroelastic response of the Goland+ wing and the Generic Business Jet Wing.

CHAPTER II

WING MESH GENERATION

The method presented in this work is a modification of mesh generation method developed by Kim and Gargloff [3, 34], which builds a hybrid mesh consisting of hexahedra and triangular prisms around a wing and allows for large out-of-plane deformations. The method divides the wing into 2D airfoil sections. A structured O-grid is built around the base airfoil section using an algebraic method with a Poisson smoother. For the rest of the domain, an unstructured mesh is constructed using a Delaunay triangulation [21]. The triangular mesh around the O-grid allows for large out-of-plane mesh deformations.

The structured O-grid is first generated for all airfoil sections. Subsequently the unstructured mesh from the base layer is generated and then mapped onto the rest of the layers. The topologically identical nodes of adjacent layers are interconnected to generate triangular prisms or hexahedra volume elements [35]. To improve the mesh quality, a tension spring analogy is applied to each layer of unstructured mesh [34].

To generate a wing mesh, a wing cap is added to the wing mesh. A Delaunay triangulation is used to mesh the interior of the airfoil section of the wingtip. The wingtip layer is then replicated from the wingtip to the boundary of the computational domain.

A distinct advantage of this method is that a complex 3D algorithm is reduced to a 2D algorithm. In addition, due to the structured nature in the spanwise direction, not only is the mesh generation simplified but the mesh lends itself to simple partitioning for parallel processing. Since the topology is identical across the layers, communication for parallel processing is simplified.

While the original meshing method is efficient, it may fail to: (1) generate high-

quality cells in the wake and at the leading edge regions of the airfoil and (2) generate a mesh around a wing/fuselage. Furthermore, the current methodology does not easily allow for chord deformations and has difficulty meshing highly cambered or deformed airfoils. However, the current methodology does allow for parallel processing and large out-of-plane wing deformations. Many of these limitations are caused by the algebraic grid generator of the O-grid around the airfoil. The first part of this chapter presents the conformal mapping method that replaced the algebraic grid generation. The second part presents the 2.5D meshing method employed in generating the 3D wing mesh.

A. 2D Mesh Generation

Conformal mapping was selected to replace the original O-grid generation method. Conformal mapping is a transformation that preserves both the size and the sign of angles. Conformal mapping makes use of the complex plane and is made up of only analytic functions within the region of interest, meaning that the derivative of the function is not zero [36, 37]. It is possible that the function is not analytic outside the region of interest. If the function is analytic a conformal map gives a one-to-one correspondence between two points [37].

The conformal mapping presented in this work is based on the work of Cizmas and Berbente [38], which uses an extension of the Carafoli method to calculate the inviscid velocity field around airfoil cascades. The work presented here uses the original algorithm to generate a 2D structured mesh around a single airfoil.

The conformal mapping algorithm consists of three analytic equations that map the exterior of the unit circle onto a cascade of airfoils. The main equation this method uses is the Weining-Manea transformation, which is shown below:

$$z = \frac{t}{2\pi}\left[e^{i\lambda}\ln\left(\frac{\zeta + R}{\zeta - R}\right) + e^{-i\lambda}\ln\left(\frac{R\zeta + 1}{R\zeta - 1}\right) + B\right] \tag{2.1}$$

where t is the cascade pitch, λ is the stagger angle, ζ is the complex variable in the mapped circle domain, and z is the complex variable in the airfoil domain. R and B are functions that depend on the solidity and the stagger angle. If the cascade pitch t is larger than 3, the influence of the neighboring airfoils is reduced and the cascaded airfoil behaves as an isolated airfoil. Using this transformation, airfoils of various thickness and curvatures can be obtained by translating and scaling the unit circle.

The conformal mapping technique traverses through three complex domains. Let $\mathscr{R}$ be the domain of the airfoil, $\mathscr{K}$ the domain of the unit circle and z and ζ the complex variables in those domains, respectively. The overall transformation function, $z_p(\zeta)$, is composed of three transformations: i) the transformation of the unit circle into the 'shaky' ellipse in $\mathscr{C}$ domain , ii) the transformation of the $\mathscr{C}$ domain into $\mathscr{E}$ domain and finally, iii) the transformation of the $\mathscr{E}$ domain into the $\mathscr{R}$ domain, by (2.1). Figure 2 shows the domains and the transformations.

The parameters necessary to calculate the $z_1(\zeta)$ and $z_2(z_1)$ are not known. To find these parameters, the inverse problem is solved, that is, from the airfoil to the unit circle. This allows any airfoil to be mapped.

The inverse of (2.1) is impossible to calculate analytically. Therefore a modified Newton-Raphson method is used to the compute the E contour [38]. Since the inverse of (2.1) has infinite solutions for each point in the $\mathscr{R}$ domain a check was implemented. One solution is a non-physical solution that maps the interior of the airfoil to the exterior of the ellipse in the E domain, while the correct solution maps the exterior of the airfoil to the exterior of the E contour. The rest of the solutions map different

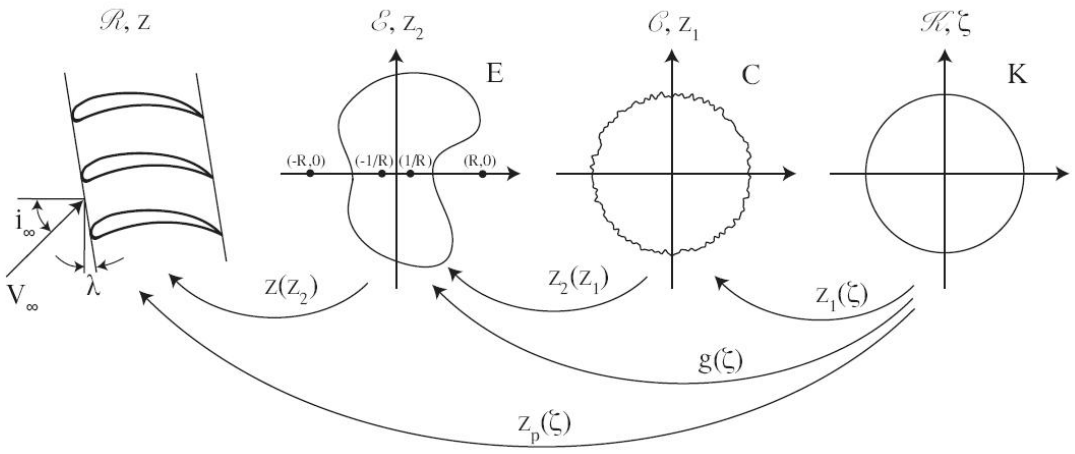


Fig. 2. Conformal mapping components.

airfoils. To ensure that the correct solution was calculated the argument of z_2 was tracked. If $\arg(z_2)$ and $\arg(z_p)$ are in opposite quadrants, then the interior solution was being calculated and the solution is discarded [37].

The shape generated by the inverse of (2.1) is similar to an ellipse. This approximate ellipse in the $\mathscr{E}$ domain is transformed into a shape closer to a true ellipse in the $\mathscr{C}$ domain using the inverse of the equation displayed below. The parameters, c^2, τ, and z_o are found by fitting an ellipse to the E contour and solving for the necessary parameters.

$$z_2 = \left(z_1 - \frac{c^2}{z_1}\right) e^{i\tau} + z_o \tag{2.2}$$

where τ is the angle between the semi-major axis of the proximal ellipse and the real axis of domain $\mathscr{E}$, $c^2 = (b^2 - a^2)/4$, where a and b are the proximal ellipse's semi-major/minor axes, and z_o is the complex coordinate of the center of the proximal ellipse [38, 37]. The calculation of the proximal ellipse is discussed in greater detail in Section 1.

The final transformation uses the equation displayed below to translate the unit circle into the C contour.

$$z_1 = \sum_{j=-1}^{m-2} C_{-j}\zeta^j \tag{2.3}$$

where the C_{-j} is the j^{th} complex constant of the transformation.

$$C_{-j} = A_{-j} + iB_{-j} \tag{2.4}$$

While ζ are the coordinates of the unit circle and therefore known, the coefficients, C_{-j} are not known. To find C_{-j}, the inverse Fourier transform is taken of (2.3). The calculation of these constants is discussed in greater detail in Section 2.

In summary, the overall transformation function $z_p(\zeta)$ shown in Fig. 2, consists of: (i) conformal mapping $z(z_2)$ given by (2.1) where ζ was substituted by z_2, (ii) conformal mapping $z_2(z_1)$ given by (2.2), and (iii) conformal mapping $z_1(\zeta)$ given by (2.4) such that the overall transformation equation is:

$$z_p(\zeta) = z(z_2) \circ z_2(z_1) \circ z_1(\zeta) \tag{2.5}$$

1. Proximal Ellipse to Contour E

The equation for an ellipse can be written as a quadric curve, which is the most useful form for finding the closest ellipse to the E contour.

$$a_e x^2 + b_e y^2 + c_e xy + d_e x + e_e y + 1 = 0 \tag{2.6}$$

Each point in the E contour, (x_i, y_i) satisfies (2.6) up to an error ϵ_i. Therefore, (2.6) can be rewritten as:

$$a_e x_i^2 + b_e y_i^2 + c_e x_i y_i + d_e x_i + e_e y_i + 1 = \epsilon_i \tag{2.7}$$

The coefficients, a_e, b_e, c_e, d_e, and e_e are defined so that the error term S is minimized:

$$S = \sum_{i=1}^{N} \epsilon_i^2 w_i = \min \tag{2.8}$$

where w_i is the weighting function and N is the number of points in the E contour.

To compute the coefficients of the quadric curve equation the derivative is taken of S with respect to each of the five parameters, a_e, b_e, c_e, d_e, and e_e, which generates a systems of equations that can be solved to find the five parameters. The system is solved using Gauss-Siedel elimination.

Once the five coefficients have been calculated, the proximal ellipse is rotated by τ where τ is equal to:

$$\tau = \tan^{-1}\left(\frac{2c}{b-a}\right) \tag{2.9}$$

The ellipse now conforms to the standard equation for an ellipse:

$$\frac{(x-x_o)^2}{a^2} + \frac{(y-y_o)^2}{b^2} = 1 \tag{2.10}$$

where a and b are the semi-major/minor axes of the rotated ellipse and are used to find c^2. x_o and y_o are the coordinates for the center of the proximal ellipse, and are found using the rest of the coefficients. An in-depth look at how the rotated axes and (x_o, y_o) were computed can be found in [39]

2. Fourier Coefficients

Equation (2.3) can be written using polar coordinates as:

$$z_1 = \sum_{j=-1}^{m-2} (A_{-j} + B_{-j}) r^{-j} (\cos j\phi - i \sin j\phi) \tag{2.11}$$

To find A_{-j} and B_{-j}, a value is assumed for ϕ by dividing the unit circle into $2m$ equal sections, where $m = 2^k k \in N$. Since (2.11) is in polar coordinates, two systems of points can be written, one even and one odd:

$$\begin{aligned} &1) \quad even: \quad \phi_{2j} = 2j\pi/m \\ &2) \quad odd: \quad \phi_{2j-1} = 2(j-1)\pi/m \end{aligned} \tag{2.12}$$

using $z_1(\phi_k) = x_k + iy_k$ and $r = 1$, for the unit circle, then (2.11) can be rewritten as:

$$\begin{aligned} x_k = \sum_{j=-1}^{n-2} A_{-j} \cos j\phi_k \quad &+ \quad B_{-j} \sin j\phi_k \\ & \qquad\qquad k = 1, 2, ...m \\ y_k = \sum_{j=-1}^{n-2} -A_{-j} \sin j\phi_k \quad &+ \quad B_{-j} \cos j\phi_k \end{aligned} \tag{2.13}$$

Since equally spaced points are used, the trigonometric functions are orthogonal to each other. This means that the coefficients A_{-j} and B_{-j} can be calculated by multiplying (2.13)a by $\cos(j\phi_k)$ and (2.13)b by $-\sin(j\phi_k)$. The summation over k results in:

$$\begin{aligned} A_{-j} = \frac{1}{m} \sum_{k=1}^{n} x_k \cos j\phi_k \quad &- \quad y_k \sin j\phi_k \\ & \qquad\qquad j = -1, 0, 1, ..., m-2 \\ B_{-j} = \frac{1}{m} \sum_{k=1}^{n} x_k \sin j\phi_k \quad &+ \quad y_k \cos j\phi_k \end{aligned} \tag{2.14}$$

To find the coefficients A_{-j} and B_{-j}, a four step iterative method is used. First the C contour is split into $2m$ equal sections with points (x_k,y_k). Half of the points correspond to the even ϕ and the other half correspond to the odd ϕ. The first step is to compute $A_{-j}^{(o)}$ and $B_{-j}^{(o)}$. The superscript number refers to which ϕ the value corresponds to, even or odd. $A_{-j}^{(o)}$ and $B_{-j}^{(o)}$ is calculated using $(x_k^{(o)}, y_k^{(o)})$ in (2.14) . For the initial iteration the points from the C contour are used. $\mathrm{A}_{-j}^{(o)}$ and $B_{-j}^{(o)}$ are then used to calculate the new ($x_k^{(e)}$,$y_k^{(e)}$) with (2.13). These points are projected onto the C contour, which generates the next set of even points, $(x_k^{(e)}, y_k^{(e)})$.

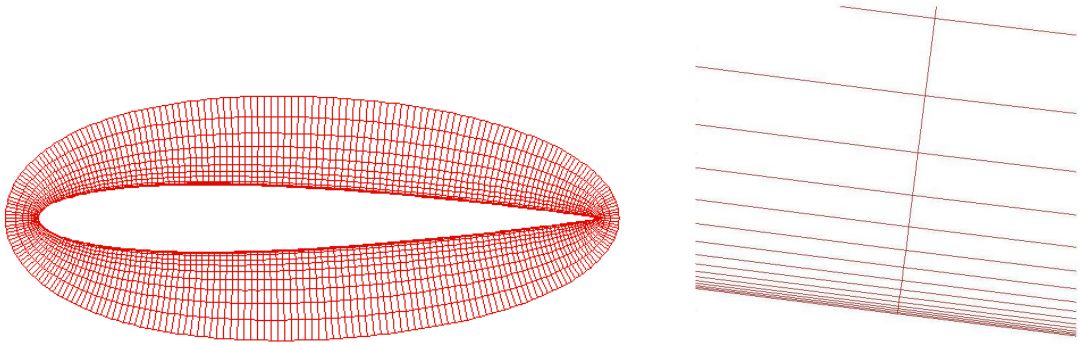
The even projected points, ($x_k^{(e)}$, $y_k^{(e)}$) are used to compute new $A_{-j}^{(e)}$ and $B_{-j}^{(e)}$, using equation (2.14). $A_{-j}^{(e)}$ and $B_{-j}^{(e)}$ are used to compute ($x_k^{(o)}$, $y_k^{(o)}$), which are projected onto the C contour to generate the new ($x_k^{(o)}$, $y_k^{(o)}$), which starts the iteration over. This iterative process continues until the distance between the calculated points and the C contour is small.

The Fourier coefficients are the last piece to complete the inverse function $z_p(\zeta)$. However, $z_p(\zeta)$ will only generate the airfoil itself. To generate the nodes for the mesh around the airfoil, the radius, r, is increased by a certain amount in (2.11) for each layer of the mesh. This results in a high quality, orthanormal quadrilateral mesh. Figure 3(a) shows a NACA 0012 mesh that was built using this method.

3. Boundary Layer

The mesh generator was designed to be used in both inviscid and viscous flow calculations. In viscous flow calculations there must be enough cells within the viscous region to adequately calculate the viscous velocity profile. In addition, the cells within the viscous region need to be high quality and preferably orthonormal. The conformal mapping satisfies these requirements by generating only orthonormal cells and allowing the user to cluster cells close to the airfoil. Figure 3(b) shows the cell clustering

around a NACA 0012 airfoil section.



(a) (b)

Fig. 3. NACA 0012: (a) conformal mapping mesh and (b) cell clustering in the viscous region.

4. Delaunay Triangulation

The Delaunay triangulation was used to fill in the domain from the outer layer of the O-grid to the outer boundary domain. The Delaunay triangulation method is implemented by program created by Shewchuk [21] called Triangle, which produces a high quality, unstructured mesh.

Triangle allows the user to define the inner and outer domain of the area to be triangulated. It also allows for "cuts "to be added within the domain. The "cuts "can be populated with a large number of cells, which allows for better computational accuracy in high disturbance areas. To help capture the wake of the airfoil correctly a cut is added from the trailing edge of the O-grid to the outlet of the domain. Figure 4 shows the outline of the structured mesh and the 'cut' used to increase mesh concentration in the wake of the airfoil.

The ability to add 'cuts' is used in the generation of the wing/fuselage mesh as well. When generating a wing mesh to use in the wing/fuselage meshing program an extra cut is added to define the where the fuselage intersects the xy plane.

Fig. 4. Wake cut.

B. Velocity Calculations

The conformal mapping equations can also be used to find the inviscid flow field around the airfoil. This is done by first calculating the flow field around the unit circle with the complex potential method [40]. The inlet and outlet boundary conditions are used to define two roto-sources, which are a combination of a vortex and a source, at the points $(R, 0)$ and $(-R, 0)$ labeled A and B in Fig. 5. To create the streamline for the unit circle, two image roto-sources at added points C and D. These roto-sources add the effects of circulation, Γ, to the flow. The potential contribution of a roto-source is well known, so we can define the potential over the field. Using the notation $\phi = \lambda - \alpha_\infty$, where α_∞ is the angle of attack, the complex potential in the K domain is:

$$F(\zeta) = \frac{t_b V_\infty}{2\pi}\left[e^{i\phi}\ln\left(\frac{\zeta-\zeta_B}{\zeta-\zeta_A}\right)\right] + e^{-i\phi} ln\left(\frac{\zeta-\zeta_D}{\zeta-\zeta_C}\right) - \frac{i\Gamma}{4\pi}\ln\frac{(\zeta-\zeta_A)(\zeta-\zeta_B)}{(\zeta-\zeta_C)(\zeta-\zeta_D)} + cons. \tag{2.15}$$

where t_b is the solidity, V_t is the reference velocity, R is the singularity point, ζ is the complex coordinates in the K domain, and Γ is the circulation. The circulation is calculated using the Kutta condition at the trailing edge.

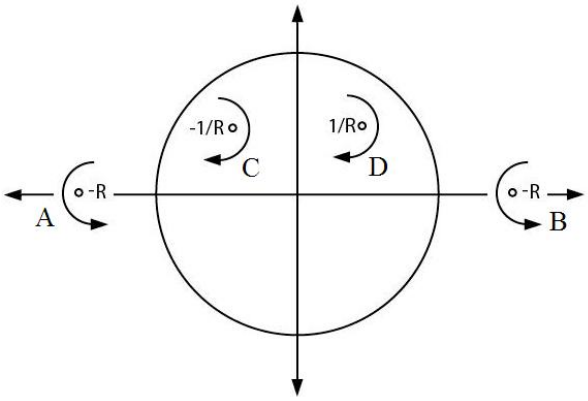


Fig. 5. Roto-sources in the K domain.

The Kutta condition requires that the velocity at the trailing edge be equal to zero. [41] To satisfy the Kutta condition, equation (2.16) is solved for Γ at $z = x_{te} + iy_{te}$. Γ is computed in the $\mathscr{K}$ domain.

$$\left(\frac{dF}{d\zeta}\right)_{te} = 0 \tag{2.16}$$

Once Γ is solved for, it is put back into (2.15) and the potential in the $\mathscr{K}$ domain is calculated. The velocity field around the unit circle is calculated by:

$$W(\zeta) = \frac{dF}{d\zeta} \tag{2.17}$$

The velocity in the $\mathscr{K}$ domain is related to the velocity in the $\mathscr{R}$ domain by equation:

$$W(\zeta) = W(z)\frac{1}{dz/d\zeta} \tag{2.18}$$

where ζ is the complex coordinate of the K domain and z is the complex coordinate of the R domain.

Using (2.15) - (2.18) the velocity field is calculated for the structured mesh around the airfoil. From the velocities, the pressure and the density are calculated using

Bernoulli's principle and the ideal gas law.

C. Wing Volume Mesh Generation

The unstructured base layer mesh is used as a source for the unstructured meshes in the rest of the domain, which ensures that each layer is topologically identical. A structured O-grid is generated separately for each layer, and then the unstructured base layer mesh is mapped onto the rest of the layers. Edges are then added between the twin nodes on each layer, creating prisms in the unstructured mesh and hexahedra in the structured mesh. This method is known as the 2.5D mapping technique [35].

1. Tension Spring Analogy

Simply mapping the unstructured mesh onto the other layers would cause overlapping cells for non-uniform wings because of the change in airfoil shape. To rectify this and improve the mesh quality, a tension spring analogy is applied to each layer of the mesh [34]. The spring analogy technique models each edge in the mesh as a spring. The network of springs adjusts the nodes iteratively until each spring is at equilibrium. The spring stiffness for an edge connecting nodes i and j is inversely proportional to the length of the edge:

$$k_{ij} = \frac{1}{\sqrt{(x_j - x_i)^2 + (y_j - y_i)^2}} \tag{2.19}$$

The updated coordinates of the nodes were found by iteratively solving the static equilibrium equations of the spring network.

$$
\begin{aligned}
F_{x_i} &= \sum_{j=1}^{N} k_{ij}(\Delta x_i - \Delta x_j) \qquad \Delta x_i = x_i^{(n)} - x_i^{(n-1)} \\
F_{y_i} &= \sum_{j=1}^{N} k_{ij}(\Delta y_i - \Delta y_j) \qquad \Delta y_i = y_i^{(n)} - y_i^{(n-1)}
\end{aligned} \tag{2.20}
$$

where F_{y_i} is the force exerted along the y axis, F_{x_i} is the force exerted along the x axis, N is the number of nodes connected to node i, Δy_i and Δx_i are the node displacements, n is the current mesh deformation step and $n-1$ is the previous mesh deformation step. F_{x_i} and F_{y_i} are both equal to zero because static equilibrium is assumed for the system to solve for the near nodes. Solving for Δx_i and Δy_i in (2.20):

$$
\begin{aligned}
\Delta x_i &= \left(\sum_{j=1}^{N} k_{ij} \cdot \Delta x_j \right) / \sum_{j=1}^{N} k_{ij} \\
\Delta y_i &= \left(\sum_{j=1}^{N} k_{ij} \cdot \Delta y_j \right) / \sum_{j=1}^{N} k_{ij}
\end{aligned} \tag{2.21}
$$

These equations are solved iteratively until the solution reaches equilibrium. This method is used repeatedly throughout this work to maintain the quality of the cells.

To fit the unstructured mesh onto the rest of the layers while still maintaining the wing shape, the O-grid and the outer boundary of the domain are held fixed as the unstructured nodes are allowed to deform as dictated by the tension springs. This allows non-uniform wings to be meshed. Figure 6 shows the source mesh layer and the wing-tip layer of the M6 ONERA wing [1]. The topology of the mesh is constant, but the unstructured mesh has been stretched to conform to the O-grid around the wing-tip airfoil size and shape.

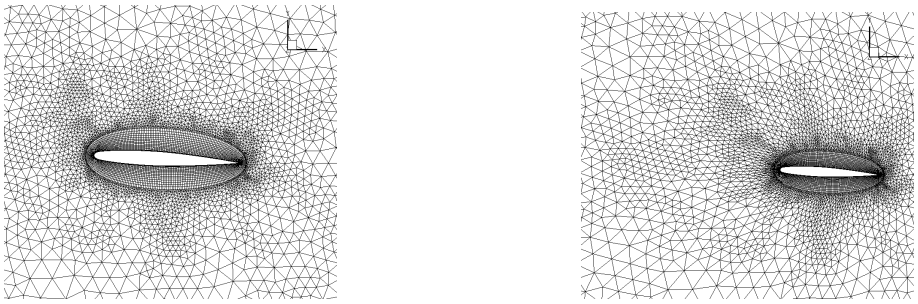


Fig. 6. The source layer (Right) and wing tip layer (Left) of the M6 ONERA.

2. Finite Wings

The 2.5D meshing method outlined in the beginning of this section can only create infinite wings. The unstructured mesh cannot be stretched enough to completely cover the wingtip airfoil section in most cases. Therefore another mesh must be generated to fill in the wingtip airfoil. This wing cap is added by using another Delaunay triangulation to mesh the interior of the wingtip airfoil section [21]. The wingtip layer is then replicated from the wingtip to the wall boundary. This creates a good approximation of a 3D wing. However, there is some loss of fidelity along the wingtip due to the fact that the layers cannot always adequately capture the curvature of the wingtip. Figure 7 shows the wing cap for the M6 ONERA wing.

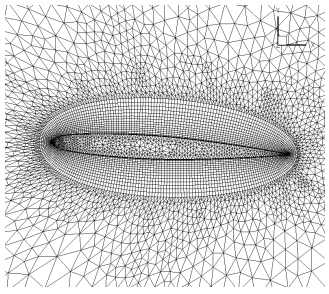


Fig. 7. M6 ONERA wing cap.

CHAPTER III

WING/FUSELAGE MESH GENERATION

This section presents a method for generating a volume mesh around a wing/fuselage. A modified version of the wing mesh generated by the algorithm in Chapter II is used as the basis for the wing/fuselage mesh. To ensure the fuselage boundary continuity, edges at the fuselage boundary are added to the unstructured mesh in the wing mesh. A surface refinement method is used to construct a 2D surface mesh on the surface of a fuselage. The fuselage background mesh is obtained from an STL file [42].

Once the surface mesh is generated, a volume mesh is generated using an algebraic method coupled with a mesh deformation technique. This volume mesh conforms to the limitations imposed by the flow solver and aeroelastic solver. The flow solver requires the mesh to be fully connected, have no overlapping cells, and have at most five sided faces [3]. Further geometric limitations are imposed due to the parallelization of the flow solver [33], which requires the mesh to have topologically identical layers. The topologically identical layers also allowed efficient use of the aeroelastic solver. These layers do not have to span the whole domain, but must include an area around the wing large enough to handle any wing deformations due to aeroelastic effects.

By using the wing mesh as basis for the wing/fuselage mesh, the advantages of the topologically identical layers are retained. Both the complex 3D surface and the 3D volume mesh are reduced to 2D problems. In addition, the structured nature of the mesh in the spanwise direction is retained and lends itself to simple partitioning for parallel processing.

A. Surface Mesh Refinement

The first step of the surface mesh refinement algorithm requires a definition for the fuselage surface, which is obtained from a sterolithography (STL) file [42]. The data from the STL file is used as a background mesh, as it is unsuitable for CFD work. For the rest of the work, the mesh obtained from the STL file will be referred to as the fuselage background mesh. More information on the STL file contents can be found on in Section 1.

Since the wing base layer mesh is planar, it must be projected onto the fuselage background mesh before the mesh refinement can begin. To ensure the best result from the projection, the unstructured wing mesh is modified by adding an additional 'cut' at the fuselage boundary in the Triangle program. The fuselage boundary is defined as the boundary where the xy plane intersects the fuselage. The fuselage boundary mesh is generated using a modified 1D advancing front method [24].

Once the fuselage boundary 'cut' has been added to the unstructured wing mesh, the wing base layer is projected onto the fuselage background mesh. To ensure accuracy and increase the speed of the method, all the nodes within the fuselage boundary are identified and labeled before they are projected onto the fuselage background mesh. To find which nodes are within the fuselage boundary, the nodes of the fuselage background mesh are forced to be co-planar with the wing base layer mesh nodes, by setting all the z-coordinates of the fuselage background mesh nodes to zero. Then, a face area method is used to check if each wing mesh node W intersects any fuselage background mesh face. If the node W does intersect a fuselage background mesh face, then the node is labeled as being within the fuselage boundary. This label is used throughout the algorithm to define which part of the mesh to refine. The area check method relies on idea that if a point, W, lies on face ABC, then the total area

of the three new faces, ABW, ACW, and BCW, will be the same as the area of ABC, otherwise the total area will be greater than the area of ABC. Figure 8(a) shows an intersecting point and Fig. 8(b) shows a non-intersecting point. The area is calculated as $\frac{1}{2}||\vec{v_1} \times \vec{v_2}||$, where $\vec{v_1}$ and $\vec{v_2}$ are the edge vectors of the triangle.

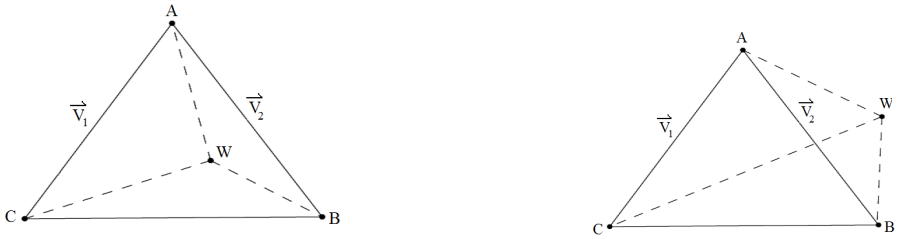


Fig. 8. Area check method: (a)intersecting point and (b)non-intersecting point.

The fuselage background mesh face that each node is projected onto is found using the method described above. Once the corresponding fuselage background mesh face is found, the new z-coordinate for the wing mesh is calculated using bi-linear interpolation, while the x and y coordinates are not modified. The projection results in the surface mesh shown in Fig. 9(a). The resulting mesh has skewed cells along the fuselage boundaries and may not capture the fuselage well. For the rest of this work, the base layer of the projected mesh will be referred to as the surface mesh, while the volume mesh around the wing/fuselage will be referred to as the wing/fuselage mesh.

Once the wing base layer mesh has been projected onto the background fuselage mesh, the mesh refinement method begins. The mesh refinement algorithm consists of a series of tessellation steps, each with four substeps: (1) node addition, (2) edge switching, (3) mesh deformation, and (4) 3D node projection onto the fuselage background mesh. Tessellation steps are repeated until a desired surface mesh refinement

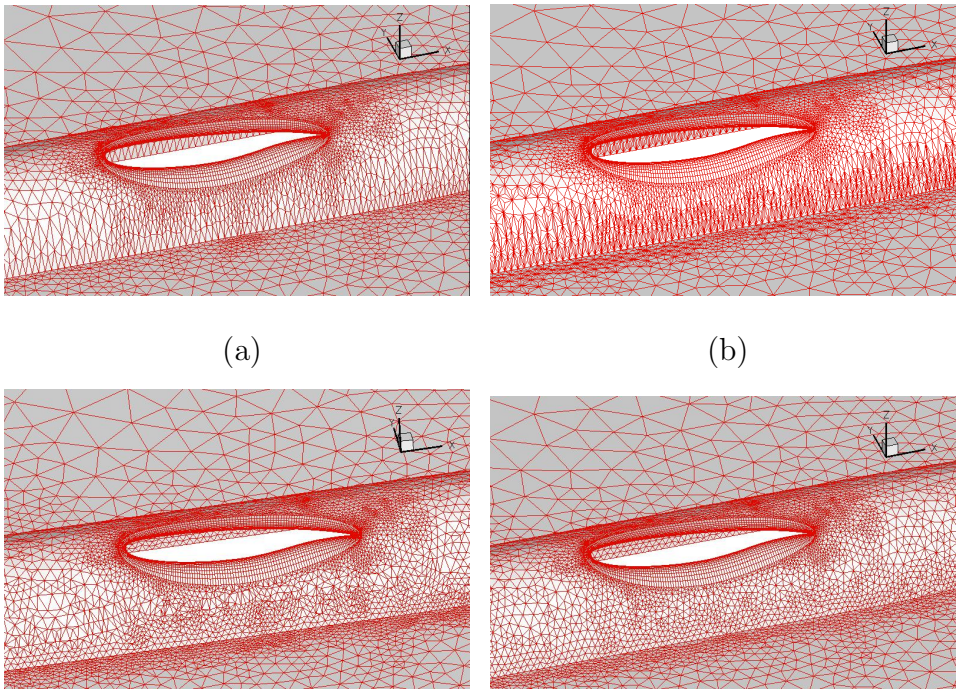


Fig. 9. Surface mesh after: (a) 2D projection, (b) node addition, (c) edge switching, (d) mesh deformation.

is reached.

The first substep, node addition, only adds nodes to the centroid of surface mesh faces within the fuselage boundary and on the boundary that meet certain criteria. A boundary face is defined as a face not on the fuselage, but attached to a face on the fuselage. The faces within the fuselage boundary were labeled in an earlier step. The criteria for adding a node are: size of the face, distance from face centroid to the fuselage background mesh, and face skewness. Each of these values is a user input value. A node is added to any large surface mesh face, any surface mesh face whose centroid is far from the fuselage background mesh, any skewed surface mesh face, and

any boundary face that has an area larger than a user defined threshold. A skewed face is defined as a face that has a much larger projected area than its original area. Figure 9(b) shows the surface mesh after the nodes are added in the first sub-step.

Adding nodes to the centroid does have some drawbacks: the added node breaks the Delaunay criterion [21], the three new faces are highly skewed, and the new node is not necessarily on the fuselage. The next three substeps, edge switching [30], mesh deformation, and 3D projection, correct these problems.

The edge switching algorithm modifies the connectivity of the surface mesh by switching the longest edge of an ill-conditioned face. An ill-condition is defined as a face with a low edge quality measure. The quality measure used is based on edge lengths and is defined by: $q_e = \frac{\sqrt{12}}{d\sqrt{p}}\sqrt{\Pi_{i=0}^{2}(p-d_i)}$, where d_i is the length of side i, $d = \max d_i$ and $p = \frac{1}{2}\sum_{i=0}^{2} d_i$. The edge quality measure, q_e, varies between 0 and 1 [29]. The highest qulaity, $q_e = 1$, corresponds to equilateral triangles. The lowest quality, $q_e = 0$ corresponds to degenerate triangles. Edge switching removes most of the highly skewed faces, improves the successes of the mesh deformation step, and improves the overall quality of the surface mesh. Figure 9(c) shows the surface mesh after the edge switching has occurred. Further details on edge switching are given in Section 2.

The third substep further improves the overall surface mesh quality by using a modified tension spring analogy. This method is explained in Section 1 on page 21 [34, 14]. The original tension spring analogy is not sufficient, because it only uses 2D springs. Therefore, it was modified to use 3D springs by adding a third equation to (2.20) and (2.21) which calculate F_{zi} and Δz_i respectively. From this the z node displacement is calculated.

The 3D tension spring analogy is applied to the whole surface mesh, however only low quality nodes are allowed to move. The quality of a node is defined as the average

edge quality of the faces connected to that node. Figure 9(d) shows the surface mesh after the tension spring analogy has been applied.

The fourth and final substep iteratively projects the new surface mesh nodes onto the fuselage background mesh, which ensures that the fuselage is captured well. Using an iterative process ensures that all the projected nodes are on the fuselage and that no skewed faces are generated. More information on the projection method can be found in Section 3.

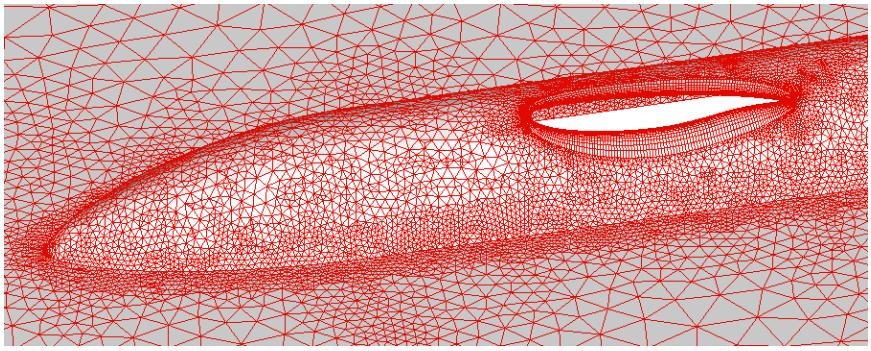


Fig. 10. Final surface mesh.

Tessellation steps are run until either a user-defined limit of iterations is reached or until no additional nodes are added to the surface mesh. At this point the surface mesh is considered complete and the rest of the wing/fuselage mesh is built. Figure 10 shows the surface mesh of the DLR-F6 fuselage [43], after four tessellations, with a minimum area of 0.005. As shown in Fig. 10, the surface mesh is refined at the intersection of the fuselage and the wall and at the nose area. This improves the discretization of the boundary layer on the fuselage.

1. STL File Format

The STL file format contains a list of triangular faces, each described by three nodes, along with a list of normals for each faces. The normals of each fuselage face are

calculated assuming that the three nodes are stored in counter-clockwise order when the face is looked down on from outside the object. However, it is possible for the nodes to be listed in the incorrect order [44], which results in a negative value normal. It is also possible for the STL file to be created with missing surfaces. To remedy this situation the STL file is first run through ADMesh, a program created to fix STL files and unify the normals. More information on ADMesh can be found in [45]. Figure 11 shows the fuselage background mesh from the STL file that was fixed by ADMesh.

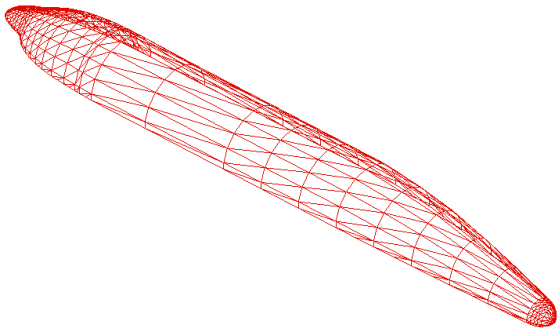


Fig. 11. Fuselage mesh from an STL file.

As shown in Fig. 11 the mesh is made up of triangles and is 'watertight'. This is what makes STL files usable as a background mesh for the mesh projection and refinement.

2. Edge Switching

The edge switching method uses the edge quality measure to determine which edges should be switched. Any triangle that has an edge quality lower than a user-defined input will have its longest edge switched with the neighboring face. To make sure no degenerate triangles are created, the edge quality and area are checked on each of the affected faces. If the total edge quality decreases after the switch, the connectivity is not changed. In addition, if the summed area of the new faces is less than the

summed area of the old faces the edge is not switched. If the summed area is larger, this usually signifies an incorrectly switched edge, or that the switch produced a skewed face. Figure 12 gives an example where edge switching improves the edge quality.

(a) (b)

Fig. 12. Edge switching: (a) before the switch and (b) after the switch.

3. 3D Projection

The 3D projection method uses an iterative process to project the nodes added in the surface mesh onto the background fuselage mesh. This method uses normals calculated at each surface mesh node for the projection. The normals at the nodes are calculated by:

$$\bar{n}_i = \frac{1}{k_i} \sum_{j=1}^{k_i} \vec{n}_j \tag{3.1}$$

where i is the indice of the node, j, is the indice of the face on the surface mesh, k_i is the number of faces adjacent to node i, and $\vec{n}_j$ is the surface mesh normal for face j [46].

It is possible that the added faces have negative normals. To ensure that all the surface mesh normals are positive, each surface mesh normal is compared to the corresponding fuselage background mesh normal. If the angle between the fuselage

background mesh normal and the surface mesh normal is greater than 25 degrees, then the order of the nodes on the surface mesh face is reversed. This process occurs before each tessellation step and before the node normals are calculated.

Once the surface mesh normal vectors have been calculated at each node, a node, $\vec{x}_i^n$, on the surface mesh is projected onto the background fuselage mesh using the following steps:

1. Calculate the distance between the surface mesh node, $\vec{x}_i^n$ and the fuselage background mesh face k

2. Find the fuselage background mesh face that corresponds to node $\vec{x}_i^n$ by projecting the node onto the plane containing fuselage background mesh face k using:

$$\vec{x}_p = \vec{x}_i^n - \vec{n}_k \cdot ds \tag{3.2}$$

   where $\vec{n}_k$ is the fuselage background mesh normal for face k and ds is the distance from the plane with face k to the node $\vec{x}_i^n$.

3. Check $\vec{x}_p$ to find if $\vec{x}_p$ intersects face k using the area method in Section A.

4. If $\vec{x}_p$ is contained within the face and ds is less than half the width of the fuselage, continue on to step 5. If ds is greater than half the fuselage width, the the node is being projected onto the interior of the fuselage. Otherwise, go back to step 1 and check another fuselage background mesh face.

5. Modify $\vec{x}_i^n$ by:

$$\vec{x}_i^{n+1} = \vec{x}_i^n - \vec{n}_i \cdot ds^n \tag{3.3}$$

   where ds^n is the distance between the node $\vec{x}_i^n$ and background fuselage mesh k, $\vec{n}_i$ is the surface mesh normal calculated using (3.1), $\vec{x}_i^{n+1}$ is the new node, and ^{n} represents the iteration number.

6. Calculate the distance, ds^{n+1} between the new node, $\vec{x}_i^{n+1}$ and the fuselage background mesh face k.

7. If ds^{n+1} is less than 1x10^{-8} then $\vec{x}_i^{n+1}$ is the new node, otherwise go back to step 5 and use $\vec{x}_i^{n+1}$ as the starting point. Continue iterating through steps 5 - 7 until ds^n is small.

The iterative method uses normals from both the fuselage background mesh and the surface mesh normals to ensure mesh quality and conformity to the fuselage shape. The fuselage background mesh normals are used to ensure that the fuselage background face closest to the surface mesh node is found. Projecting the surface mesh nodes with the surface mesh normals ensures that no skewed cells are created. However, using the surface mesh normals requires the iterative process detailed above to find the correct distance to the fuselage background mesh. Figure 13 shows each of the vectors and nodes discussed above. The angles and distances are exaggerated for illustrative purposes.

B. Volume Meshing

The volume mesh for the wing/fuselage is generated using an algebraic method coupled with a mesh deformation method. The final wing/fuselage volume mesh consists of topologically identical layers in the span direction with tetrahedra cells around the wing and hexahedra cells for the rest of the domain. The base layer of the final wing/fuselage mesh is the surface mesh generated during the surface mesh refinement process.

The first step in this process is part of the surface mesh refinement. To keep all the layers topologically identical, every node added in the mesh refinement algorithm to the the surface mesh is added to the rest of the layers. In addition, as the connec-

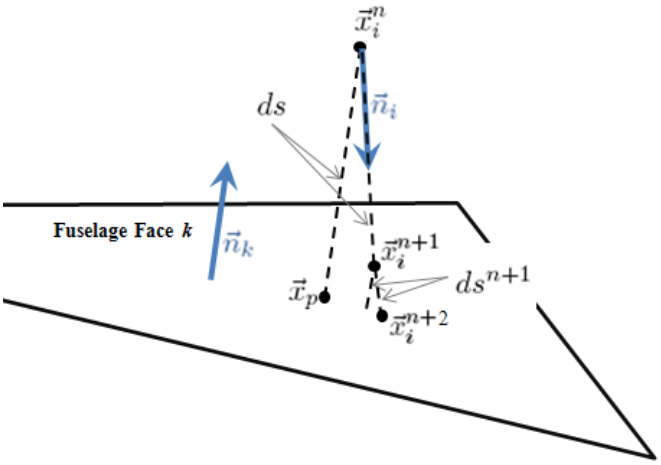


Fig. 13. Surface mesh node projection.

tivity of the surface mesh is modified, the connectivity for the rest of the layers are modified as well.

To account for the fuselage shape in the wing/fuselage mesh, the z-coordinates of all the wing mesh layers except the base layer must be modified. Each layer is assumed to lie on the plane $z = 0$ and then the layers are projected onto the fuselage background mesh using the projection method described in Section A on page 25. An algebraic distribution method is then used to generate the final z-coordinate of the volume mesh for each layer:

$$z_{i,l} = \Delta z_l + z_{i,l}^p \left(1 - \frac{\Delta z_l}{z_{max}}\right) \tag{3.4}$$

where the subscript l refers the layer number, the subscript i refers to the node, Δz_l desired distance between each layer, $z_{i,l}^p$ is the projected z-coordinate, and z_{max} is the height of the last layer. This distribution results in the fuselage being fully formed in the base layer and the final layer being flat, with coordinates of $z = z_{max}$.

As noted earlier, the projection can produce low quality cells. To remedy this, the tension spring analogy is run on each layer starting with the layer closest to the surface mesh. To maintain the shape of the fuselage within the wing/fuselage mesh, each node is attached to its twin on the layer below it. The node on the lower layer is considered fixed as the tension spring analogy is run on each layer, starting with the layer closest to the surface mesh and continuing upwards along the span. Only low quality faces are allowed to deform.

Since the layers have been redistributed along the span, the x and y coordinates of the wing/fuselage mesh layers maybe incorrect. The final x and y coordinates of the mesh are calculated from the wing mesh by using a simple linear interpolation. The interpolation is done by finding the two original wing mesh layers the new z-coordinate falls between and using the wing mesh x and y coordinates to calculate the new x and y coordinates.

If more layers are needed to capture the boundary layer, then layers can be added in between the already existing layers. This is also done using linear interpolation. This allows the user to change the number of layers in the final mesh without completely starting over.

CHAPTER IV

RESULTS

A. Grid Quality

This section presents the results for the quality of both the two-dimensional and three-dimensional grid generation methods. The first part compares the grid quality of a conformal mapping grid against an algebraic mesh grid with Poisson smoothing (APS grid). The second part of this section discusses the mesh quality for the three-dimensional fuselage mesh.

1. Wing Mesh Quality

The quality of the two-dimensional grid produced by the conformal mapping mesh generator was compared with the quality of the APS grid [34]. The quality of the unstructured mesh was investigated in [34] and is not presented here. Two indicators are proposed to evaluate the quality of the mesh: one based on edge anles and a second based on the aspect ratio. These indicators assess the mesh quality of the layers perpendicular to the wing span direction. Consequently, the three-dimensional grid quality problem is reduced to two dimensions.

The first indicator depends edge angles and is defined as:

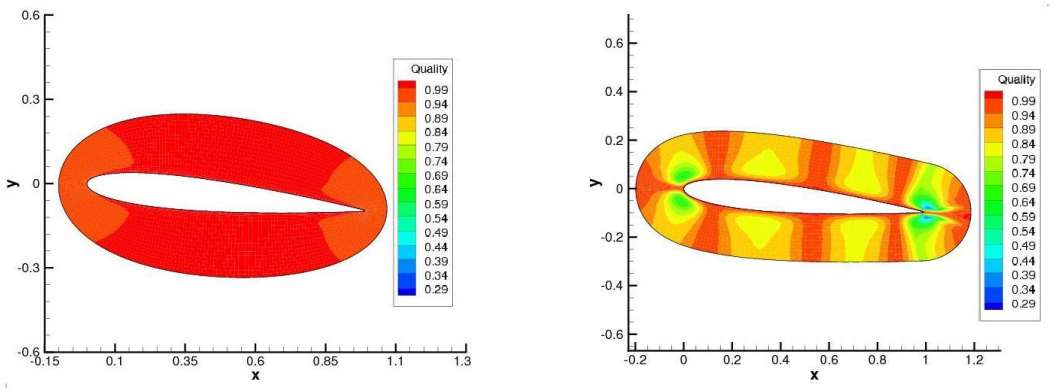
$$q_i = 0.25 \sum_{i=1,4} \sin \alpha_i \tag{4.1}$$

where α_i is the angle at node i. The mesh quality improved as the indicator approached 1, the maximum value of q. This quality measure will be referred to as angle quality.

The second indicator of mesh quality is the face aspect ratio. The aspect ratio

is defined as: $AR = \frac{area_i}{\ell_i^2}$ where ℓ_i is the longest side of cell i. The aspect ratio of a cell defines how skewed the cell is. As face skewness increases, the aspect ratio decreases [47].

To compare the quality of the APS grid to the conformal mapping grid, a grid was constructed using each method. On both grids the $y+$ numbers were matched and the number of nodes was kept within 10%. The O-grids were generated about a NACA 0012 airfoil at 5.56° angle of attack. Figure 14 indicates that the vertex quality of the conformal mapping grid exceeds that of the APS grid. It should be noted that, although the conformal mapping generates an orthogonal grid, the edges of the cells are approximated by straight lines and therefore the average angle was not 90°.



(a) (b)

Fig. 14. Vertex-based grid quality: (a) conformal mapping and (b) APS mesh.

Table I provides the mesh quality measures for each grid. The average and minimum angles were calculated from the quality indicator, q as: $\theta = \arcsin(q)$

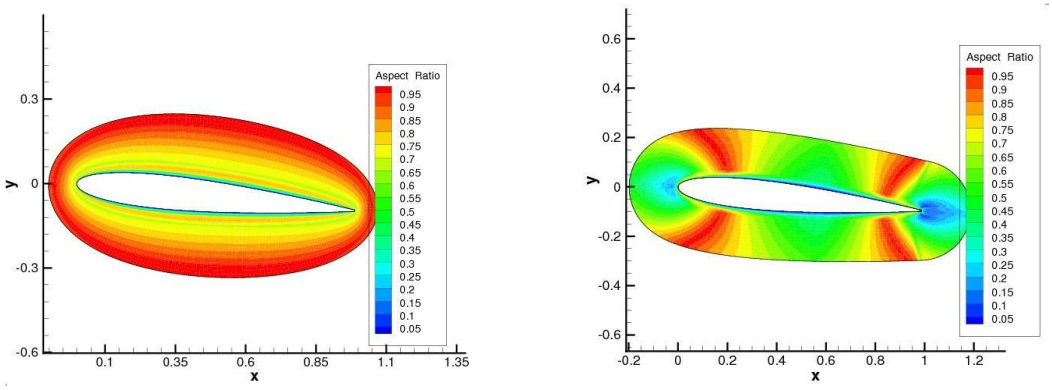
Table I confirms that the conformal grid has a better angle quality measure than the APS grid. The lowest vertex quality measure for the conformal mapping grid

Table I. APS and conformal mapping grid quality.

Grid Type	O-grid Nodes	Max y+	Avg Angle Quality	Avg Angle	Min Angle	Avg Aspect Ratio
APS	6,275	1.705	0.899	64.0^o	17.5^o	0.45
Conformal	6,912	1.675	0.987	80.9^o	59.7^o	0.46

generation method is at the leading and trailing edges. The lowest angle quality measure for the APS grid is located at the trailing edge, within the wake region.

Figure 15 provides a comparison of the aspect ratios for each grid. The conformal mapping grid generation method yields low aspect ratio faces in the wake region of the flow. Both methods produce low aspect ratio cells within the viscous layer due to limitations of the number of nodes that can be distributed around the airfoil.



(a) (b)

Fig. 15. Aspect ratio-based grid quality: (a) conformal mapping grid and (b) APS grid.

2. 3D Fuselage Mesh Quality

The mesh edge quality indicator and the methodology for building the wing/fuselage mesh were described in Chapter III. The methodology was broken down into major tessellation steps, containing minor mesh refinement steps within the tessellation step. At each minor step the quality of the three-dimensional unstructured mesh was calculated. The quality of the structured mesh was not calculated as the connectivity of the structured mesh was not modified. Fig. 16 shows the average mesh edge quality for each minor step within each tessellation step, for a three tessellation step process.

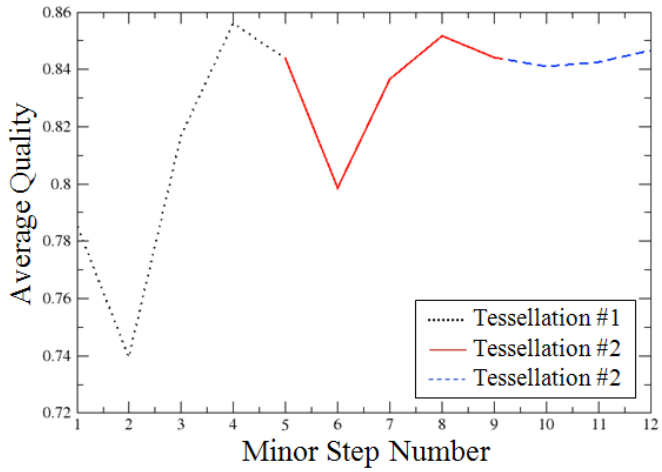


Fig. 16. Tessellation step: cell quality.

As shown in the figure, the first minor step reduces the overall edge quality of the mesh. The reduction in edge quality at each second minor step is due to the addition of nodes to the mesh. The purpose of the remaining minor steps is to refine the mesh connectivity and node placement to improve the mesh edge quality. The nodes are added to improve mesh conformity to the fuselage. As the tessellation steps progress, the fluctuations in the mesh edge quality decrease, until settling to a cell

edge quality of 0.86. Figure 17 illustrates the edge quality distribution of the cells in the final mesh. Approximately 98% of the cells have an edge quality of 0.6 or greater, with the greatest concentration of cell within the 0.75 - 0.95 range. No cell has an edge quality less than 0.25.

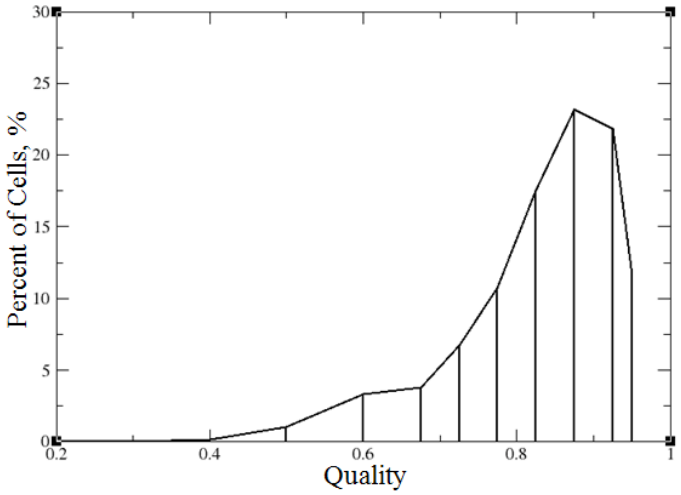


Fig. 17. Cell edge quality distribution.

B. NACA 0012

This section presents the simulation of the viscous turbulent flow over a NACA 0012 airfoil. The simulation was used to evaluate the effect of the conformal mapping grid generation on a viscous flow simulation over an infinite wing and to compare the APS grid with the conformal mapping grid. The flow solver was a Reynolds-averaged Navier-Stokes solver that used a k-ω turbulence model. The flow solver was previously validated in [3]. The numerical results from the flow solver were compared to the experimental data from Harris [48].

The mesh was generated with the inlet and outlet boundaries located 10 chord

lengths away from the center of the airfoil. Additionally, the upper and lower boundaries were also located 10 chord lengths away from the center of the airfoil. The airfoil was rotated to be at an angle of attack of 5.58°. Three conformal mapping meshes were constructed to verify that the solution is grid independent. To evaluate the effect of the conformal mapping grid generation method on the viscous flow simulation, an APS grid was also contructed [34]. The APS grid used the same coordinates around the airfoil as the medium sized conformal mapping grid. The parameters for each mesh are provided in Table II, where I_{max} denotes the number of nodes around the airfoil and J_{max} denotes the number of O-grid layers. Total nodes include both the unstructured and structured meshes.

Table II. NACA 0012 mesh parameters.

Grid Type	Grid Size	I_{max}	J_{max}	Total Nodes	y+ Number	C_L	C_D
Conformal	Coarse	128	20	9,272	0.6121	0.359	0.0112
Conformal	Medium	256	25	22,656	0.5687	0.398	0.0605
Conformal	Fine	512	45	60,450	0.5976	0.397	0.0507
APS	Medium	256	25	19,286	0.5784	0.404	0.0659

The flow conditions for the test case were: P^*_{inlet} = 107,853 Pa, P_{exit} = 101,325 Pa, and T_{inlet} = 300 K, which resulted in a Mach number of 0.3. The airfoil had a no penetration boundary condition and a no-slip boundary condition. The flow solver was run until the average residual error was less than 1x10^{-7}. The log of the average residual for all five flow parameters is shown in Fig. 18.

To confirm solution grid independence, the lift and drag coefficients were calculated for all four grids and are displayed in Table II. The medium and fine grids had little variation in the lift and drag coefficients. The error between the two lift

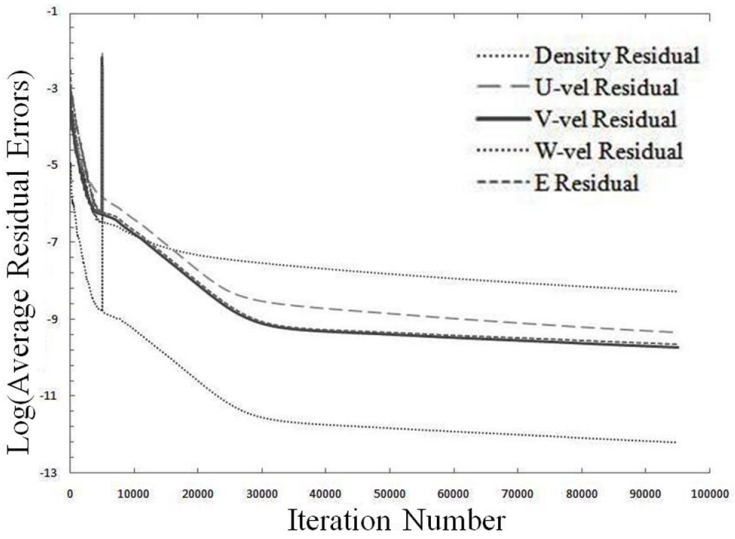


Fig. 18. NACA 0012 medium grid average residual errors.

coefficients was 0.2%. This means the solution was grid independent for grids larger than 22,000 nodes. As a result, the medium grid was used to compare the APS grid to the conformal mapping grid solutions as it provided the best compromise between accuracy and computational efficiency. Also included in the Table II is the lift and drag coefficent for the APS grid. The average error between the medium conformal mapping grid and the APS grid was slight, with an error of about 0.7%.

To further assess the effect of the conformal mapping grid on the viscous solutions, the experimental pressure coefficients were compared with the numerical pressure coefficients from the flow solver. Figure 19 provides the plot of the pressure coefficients for both the experimental and the numerical data. All three conformal mapping meshes and the APS grid solutions are included in Fig. 19. To quantify the quality of the results, the error was defined as: $\epsilon_i = (P_{numerical} - P_{experimental})/P_{experimental}$, where P is the pressure. Table III shows the average error and the max error for all four grids. Both grids generated quality results, which suggests that the conformal mapping grid has little effect on the solution in this test case.

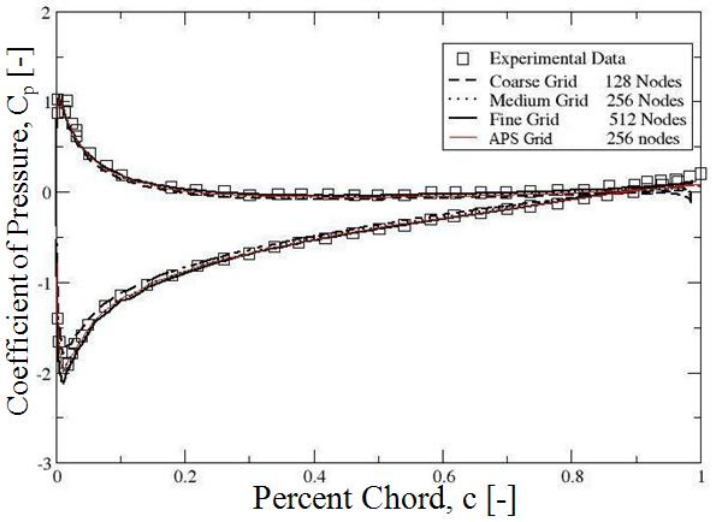


Fig. 19. Pressure coefficient variation: experimental data and RANS solver results.

C. M6 ONERA

This section presents the numerical results for the simulation of viscous flow around the M6 ONERA wing platform [49]. This simulation was conducted to examine the effect of the conformal mapping grid generation for a transonic viscous flow over a finite wing. The M6 ONERA simulation was used to compare the results from the APS grid against solutions from the conformal mapping grid. The flow was predicted using a Reynolds-averaged Navier-Stokes solver. The results from the flow solver were compared with experimental results from [49].

The mesh was constructed with inlet and outlet boundaries located ten chord lengths away from the center of the airfoil. Additional, the upper and lower boundaries of the domain were also located ten chord lengths away from the center of the airfoil. The base layer of the medium sized mesh from the grid convergence study, along with the wing surface mesh, can be seen in Fig. 20. The grid convergence study was done using four grids for which the O-grid was discretized using conformal

Table III. NACA 0012 error.

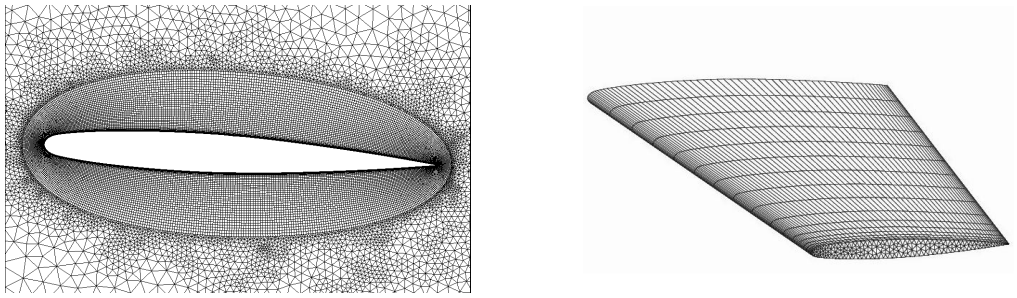
Grid Type	Grid Size	ϵ_{ave}	ϵ_{max}
Conformal	Coarse	2.02%	13.73%
Conformal	Medium	1.81%	12.8%
Conformal	Fine	1.88%	12.07%
APS	Medium	2.34%	13.14%

mapping. A fifth grid was generated using the APS grid methodology, with the same wing surface grid coordinates as the medium sized conformal mapping grid. The mesh parameters for all five meshes are displayed in Table IV.

Table IV. M6 ONERA wing mesh parameters.

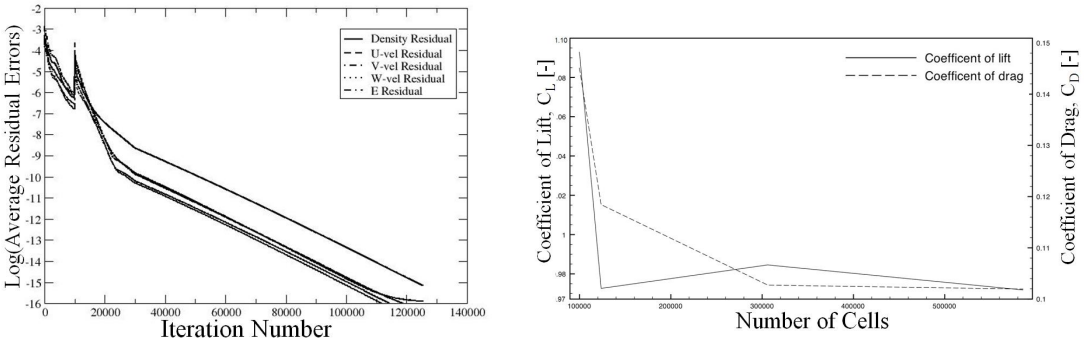
Grid Type	Grid Size	I_{max}	J_{max}	K_{max}	Total Nodes	y+ Number
Conformal	Very Coarse	128	21	13	159,271	2.16
Conformal	Coarse	256	23	21	270,813	1.72
Conformal	Medium	256	52	23	490,363	1.69
Conformal	Fine	512	55	24	847,008	1.48
APS	Medium	256	52	23	587,346	2.09

The flow conditions for the M6 ONERA wing were: P^*_{inlet} = 160,750 Pa, T^*_{inlet} = 293 K, and P_{exit} = 101,325 Pa. The Reynolds number was Re = 24,288,000 and the angle of attack was 3.06°. The simulation was run until the average residual errors were less than 1x10^{-11}. The plot of the errors can be seen in Fig. 21(a). The spike of the residuals at 10,000 iterations is due to switching from first-order spatial accuracy to second-order spatial accuracy.



(a) (b)

Fig. 20. M6 ONERA mesh: (a) base mesh and (b) wing surface mesh.



(a) (b)

Fig. 21. M6 ONERA: (a) average residuals and (b) grid convergence.

Figure 21(b) shows the lift and drag coefficients vs. O-grid size for the four conformal mapping grids. I_{max}, J_{max}, and K_{max} are the nodes around the airfoil, the O-grid layers and the span layers respectively. As Fig. 21 shows, the lift and drag coefficients are independent of grid size when the mesh exceeds 490,000 nodes. As a result, the medium grid was the best compromise between accuracy and computational efficiency.

The plots of the pressure coefficient for four span locations can be seen in Fig. 22. This flow regime causes an interesting double shock feature on the suction side of the wing. In both the conformal mapping grid and the the APS grids, there was an oscil-

lation in the pressure prior to second shock that was not present in the experimental results. The contour plot for the 65% span location is shown in Fig. 23. Both the double shock and the numerical oscillation can be seen in Fig. 23. This numerical oscillation is believed to be caused the equal spacing of the points around the airfoil. Further research into this numerical oscillation is being done.

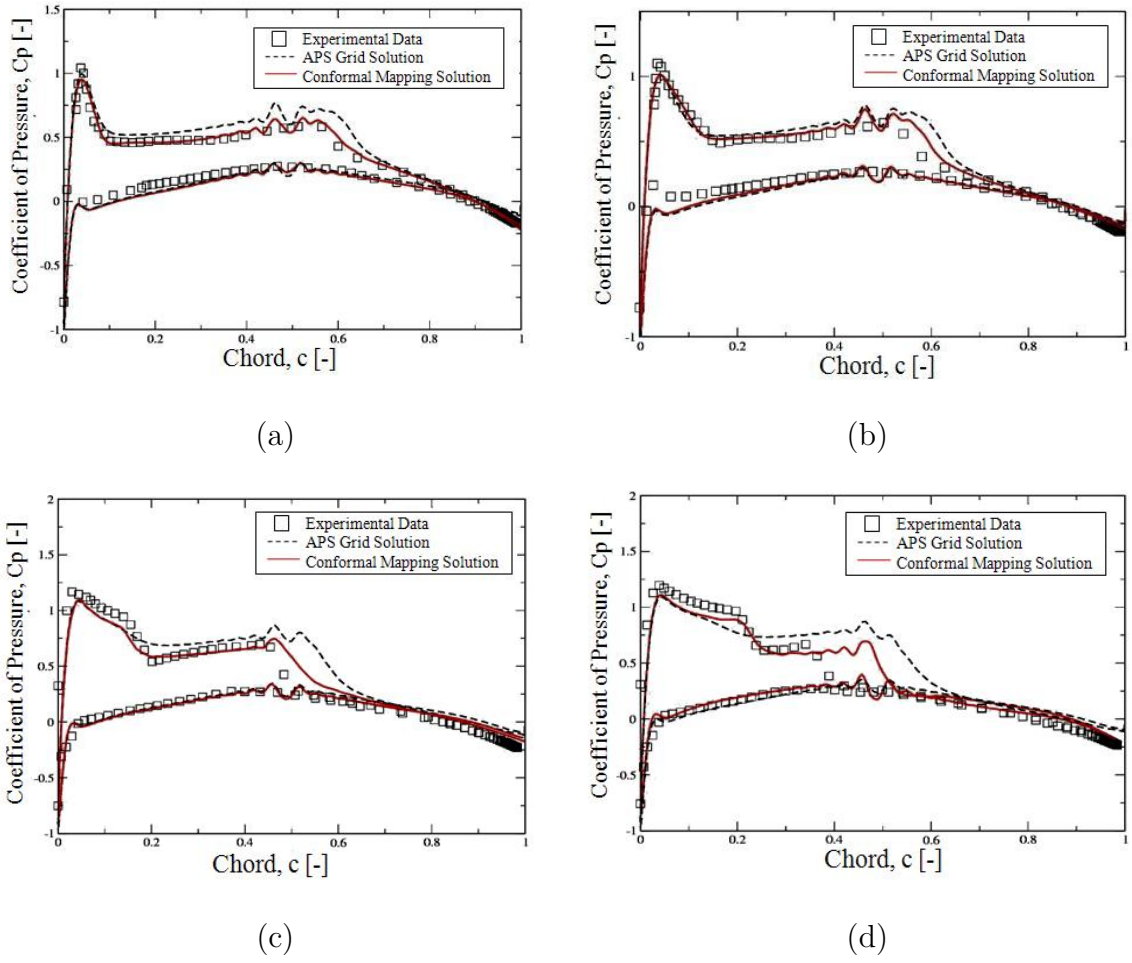


Fig. 22. Pressure coefficient variation: experimental data and numerical results at (a) 20%, (b) 33%, (c) 65% and (d) 85% span.

As Fig. 22 shows, the conformal mapping grid more accurately captures the flow

features than the APS grid. At 85% span, the average error, ϵ_{ave}, for the conformal mapping grid was 2.3%, while the APS grid produced an ϵ_{ave} of 6.3%. The maximum error, ϵ_{max}, for both grids is at the second shock location. The conformal mapping grid produced an ϵ_{max} of 20.6%, while the APS grid produced an ϵ_{max} of 52.1%. These errors are due a couple of factors: (1) the wing tip grid is not refined enough and (2) the simulation was run assuming laminar flow. The wingtip grid is refined as much as possible with the current methodology. A work around is being looked into at this time. In addition, turbulent simulations are being run at this time.

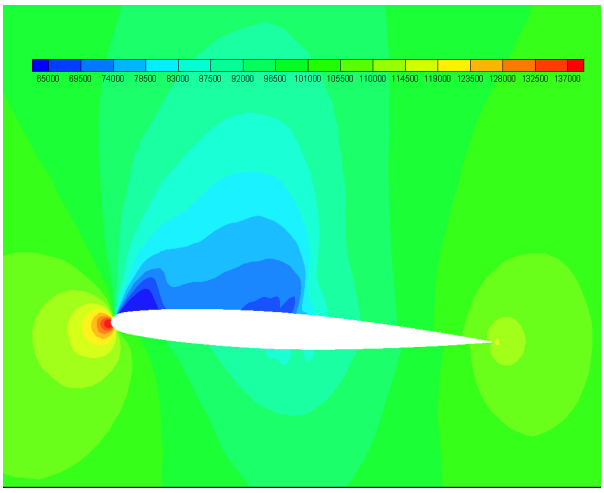


Fig. 23. M6 ONERA pressure contour.

D. DLR-F6

The DLR-F6 wing/fuselage test case was used to verify the wing/fuselage meshing algorithm proposed in this work. The DLR-F6 configuration was a twin-engine, wide body aircraft [43]. This configuration was tested in both the 2^{nd} and 3^{rd} AIAA CFD Drag Prediction Workshop. Two different configurations were run at the workshops: one with a fairing added to the wing-body interface and one without. Both of these

test cases were run in a clean body configuration (no engines attached) [43]. Herein, only the clean configuration without the fairing was considered. The flow was predicted using a Reynolds-averaged Navier-Stokes solver with a two equation $k - \omega$ shear stress transport (SST) turbulence model. The results from the flow solver were compared with results from [50].

The mesh was constructed with inlet and outlet boundaries located 10 chord lengths away from the center of the airfoil. The upper and lower boundaries were also located 10 chord lengths away from the center of the airfoil. The planform for the DLR-F6 is shown in Fig. 24. The detailed view of the wing surface mesh, along with the full wing/fuselage surface mesh, are shown in Fig. 25. A grid convergence study was done using three different meshes, whose parameters are provided in Table V. All three meshes has O-grids generated using the conformal mapping method. The lift and drag coefficients for each mesh are given in Table V. The medium grid was considered a good compromise between accuracy and computational efficacy and was used for the remainder of the calculations.

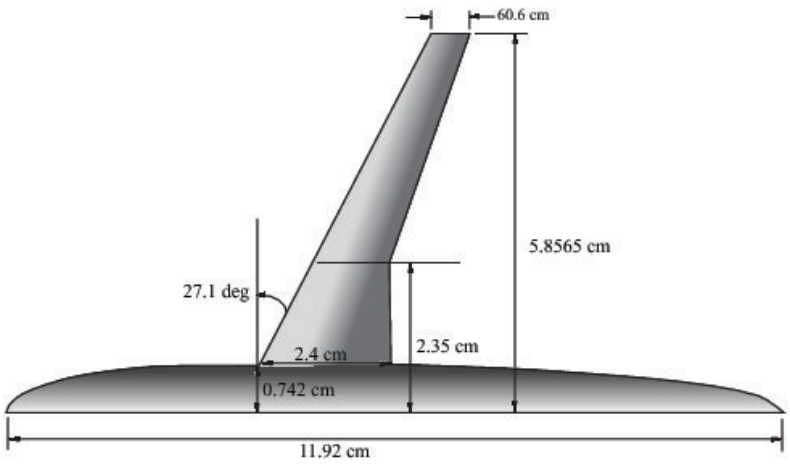
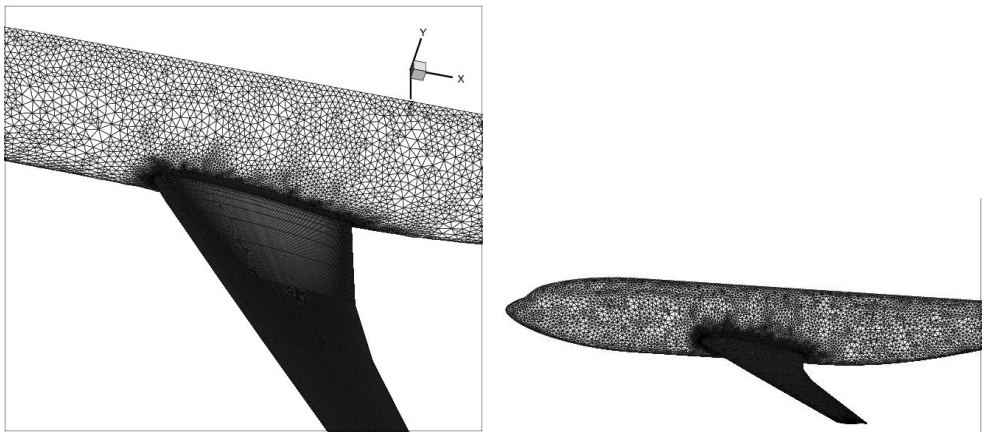


Fig. 24. DLR-F6 planform.

The flow conditions for the DLR-F6 test case were: P^*_{inlet} = 148,555 Pa, T^*_{inlet}



(a) (b)

Fig. 25. DLR-F6 surface mesh: (a) detail of wing/fuselage intersection and (b) the whole fuselage.

= 322.22 K, and P_{exit} = 101,325 Pa, which resulted in an Mach number of 0.76 and a Reynolds number of 5x10^6. The wing was set to an angle of attack of 0.719°. The flow simulation was run until the average residual errors was less than 1x10$^{-6.5}$. The plot of the log of the average residual errors is shown in Fig. 26.

Table V. DLR-F6 mesh parameters.

Grid Size	I_{max}	J_{max}	K_{max}	Total Nodes	C_L	C_D
Coarse	128	14	19	276,958	0.5320	0.122
Medium	256	22	22	452,625	0.5021	0.0951
Fine	512	42	31	1,275,958	0.5005	0.1074

Figure 27 shows the pressure coefficients for three span locations along the wing. At all three span locations, there is a numerical oscillation in the numerical results, which are at the same chord location as the numerical oscillations in the M6 ONERA

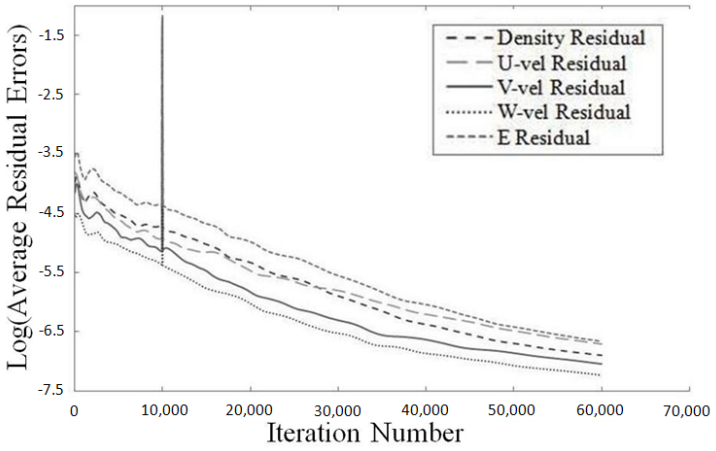


Fig. 26. DLR-F6 medium grid average residual errors.

simulation. Additional study into this phenomena is being done. At 15% wing span the max error between the numerical data and the experimental data was 11.8%, and ϵ_{ave} was 4.85%. At 64%, the mesh produces an ϵ_{max} of 30.1% at the leading edge, and an ϵ_{ave} of 6.22%. The largest error is at the mid section of the wing. These errors are mainly caused by 2 factors: (1) lack of layers in the mid section of the wing and (2) the wingtip mesh is not refined enough. A work around for the wingtip mesh in being researched now.

E. Goland+ Wing

The Goland+ wing was used to assess how the conformal mapping mesh generation algorithm proposed herein affected the results of an aeroelastic simulation. The Goland+ wing is a heavier version of the Goland wing that was proposed by Eastep and Olsen [5] for transonic flutter simulations. The Goland+ wing is rectangular, cantilevered wing with constant mass and stiffness properties. The wing dimensions are shown in Fig. 28. The flow and beam were simulated using an Euler solver, coupled with a nonlinear beam solver [33]. The results of the aeroelastic simulation were

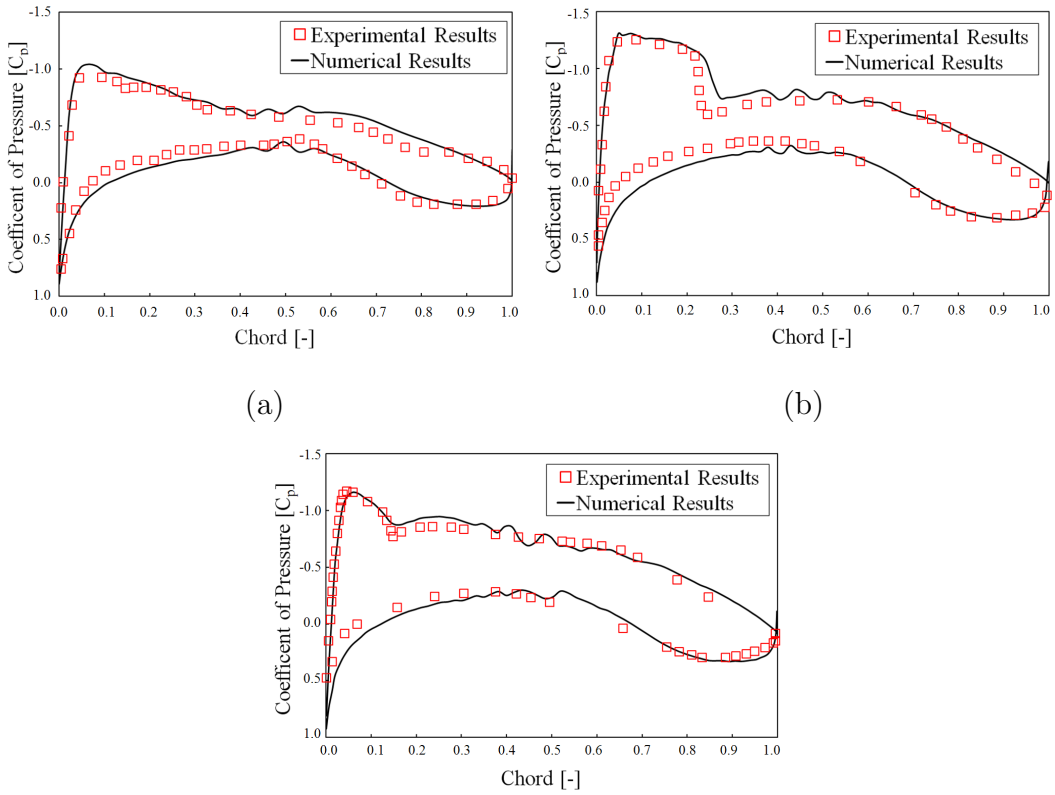


Fig. 27. DLR-F6 Pressure coefficient variation: experimental data and RANS solver results at (a) 15%, (b) 33%, and (c) 64%.

compared to numerical results from [51].

The mesh was generated with inlet and outlet boundaries located 10 chord lengths away from the center of the airfoil. The upper and lower boundaries were also located 10 chord lengths away from the center of the airfoil. The airfoil had zero angle of attack. The base layer of the mesh, along with the wing surface mesh, can be seen in Fig. 29. To check for solution grid independence, three different grids were generated. The flow was simulated using an Euler solver. The results of the grid independence investigation are provided in Table VI. The solution is grid indepen-

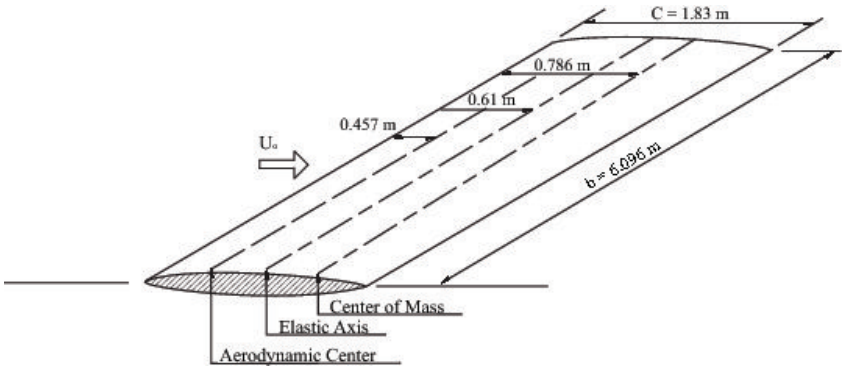
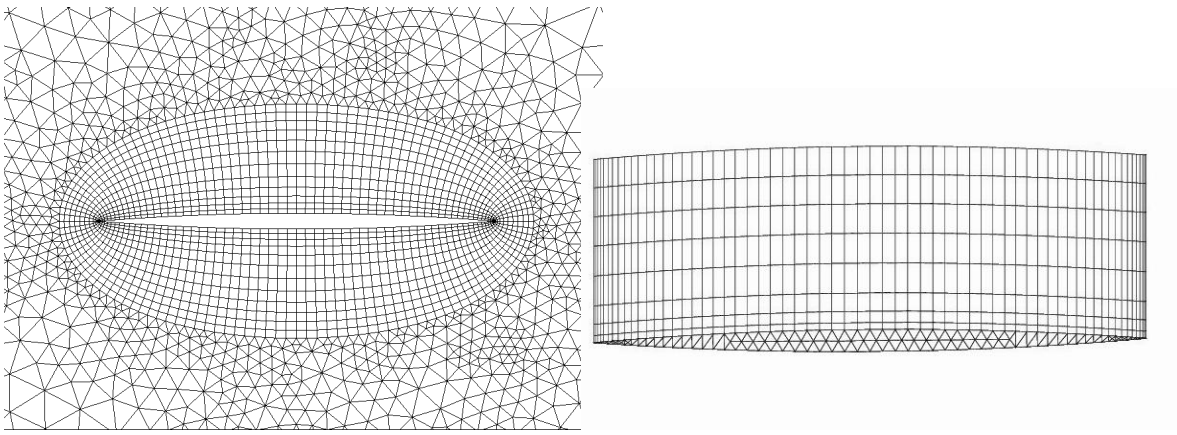


Fig. 28. Goland wing layout.

dent when the number of grid points exceeds 72,000. Therefore, the medium grid was chosen as the best compromise between accuracy and computational efficiency.

To test the effect of the conformal mapping on an aeroelastic simulation, three more grids were built using the APS grid generation methodology. These grids had the same airfoil points as the conformal mapping grids and were also tested for grid independence. The results of this test are shown in Table VI. The solution is grid independent when the the number of grid points exceeds 75,000. The parameters for the APS grids are also listed in Table VI.

The aeroelastic simulation was run using an aeroelastic solver that coupled a structural solver [52] that modeled the wing as a non-linear beam and a flow solver that modeled the unsteady aerodynamics using an Euler solver. The simulation was first run using only the flow solver to get an initial flow field for the aeroelastic solver. The flow conditions were: P^*_{inlet} = 59,774 Pa, P_{exit} = 34,578 Pa, T^*_{inlet} = 288 K, and R_{gas} = 97.951 N m/kg. These initial conditions were used to match the Reynolds number of 15x10^6 and the dynamic pressure, Q_f, of 7,170 Pa (150 psf). The resulting Mach number was 0.92. For the aeroelastic simulation, the wing was given an initial torsional deformation, at the wing tip, of 0.042 rad upwards. The Goland+ wing was



(a) (b)

Fig. 29. Goland+ wing mesh (a) base layer and (b) wing surface mesh.

modeled as a non-linear beam with the same structural properties as the experimental wing. The structural properties are presented in [51] and are not repeated here. For the aeroelastic simulation, the beam was modeled as having only two natural frequencies: 1.69 Hz and 3.05 Hz. These natural frequencies matched the frequencies from [51].

The aeroelastic simulation was run for about 4.5 seconds using the medium conformal mapping mesh and for about about 6.0 using the APS grid. Figure 30(a) shows the first modal coordinates, out-of-plane plunging mode, of the beam and Fig. 30(b) shows the second modal coordinates, pitching mode.

As shown in Fig. 30, the conformal mapping grid predicts the onset of the LCO behavior, while the APS grid predicts flutter. The LCO behavior demonstrated by the conformal mapping mesh is in agreement with results from [51]. The predicted frequency from [51] was 3.37 Hz. The conformal mapping solution had two frequencies of 3.38 Hz and 2.71 Hz, which results in an error of 0.2% error. The APS grid, however, predicts the activation of numerous frequencies: 2.32 Hz, 2.15 Hz, 3.04 Hz, 4.66 Hz,

Table VI. Goland+ grid parameters.

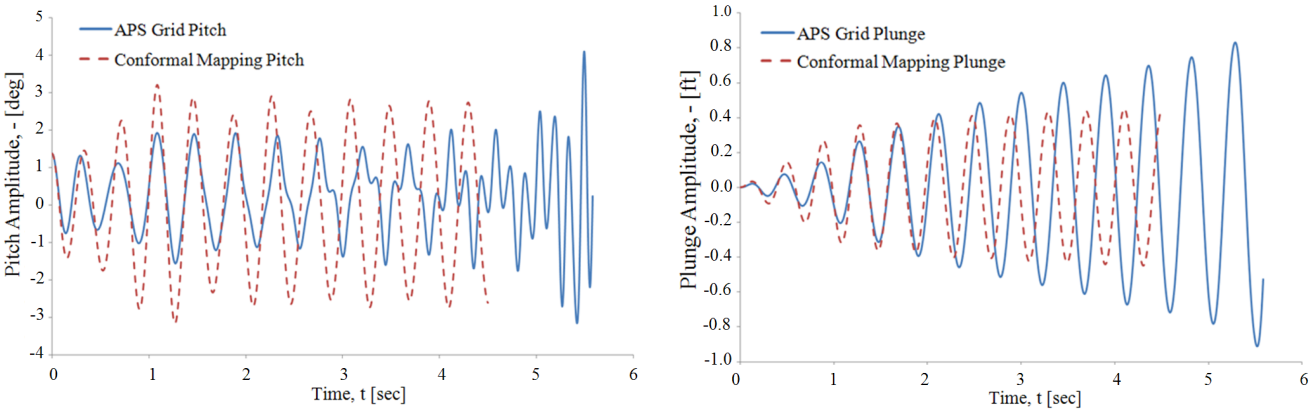
Grid Type	Grid Size	I_{max}	J_{max}	K_{max}	Total Nodes	C_L	C_D
Conformal	Coarse	128	13	16	55,480	0.0163	7.12e-3
Conformal	Medium	128	25	18	72,912	0.0163	6.92e-3
Conformal	Fine	256	35	21	168,679	0.0164	6.79e-3
APS	Coarse	128	10	16	37,336	0.0161	7.11e-3
APS	Medium	128	25	18	75,264	0.0164	7.26e-3
APS	Fine	256	24	20	188,490	0.0163	7.16e-3

and 9.49 Hz. The freqency plot for the APS grid is shown in Fig. 31.

As indicated by both figures, the conformal mapping grid is a better mesh for capturing aeroelastic effects. The conformal mapping grid correctly predicts the LCO behavior of the Goland+ wing while the APS grid does not. The greater solution accuracy is due to the ability of the conformal mapping grid to better capture the oscillating shock on the upper surface of the Goland+ wing.

F. Generic Business Jet Wing

The Generic Business Jet Wing was used to further assess how the conformal mapping mesh generation algorithm proposed herein affected the results of an aeroelastic simulation. The Generic Business Jet Wing flutter model is a semi-span model of a typical business jet wing and is constructed of a stepped thickness aluminum plate that is covered with balsa wood to provide the wing contour. Figure 32 provides the layout of the Generic Business Jet Wing. The Generic Business Jet Wing model has been previously tested in Langley's Transonic Dynamics Tunnel (TDT) in 1993 and 1994 [1]. Three different wingtip configurations were tested: clean wingtip, pencil



(a) (b)

Fig. 30. Goland+ modal coordinates: (a) out-of-plane plunging mode (b) pitching mode.

tipstore, and winglet.

The mesh was constructed with inlet and outlet boundaries located 10 chord lengths away from the center of the airfoil. Additionally, the upper and lower boundaries were also located 10 chord lengths away from the center of the airfoil. The wing had an initial angle of attack of 0.6°. The base layer of the mesh, along with the wing surface mesh, can be seen in Fig. 33. To verify solution grid independence, three different grids were generated. The flow was simulated using an Euler solver. The results of the grid independence check can be seen in Table VII. The medium grid was chosen as the best compromise between accuracy and computational efficiency. A fourth mesh was generated using an algebraic grid with Poisson smoothing for the O-grid. This grid was used to test the effect of the conformal mapping grid on the aeroelastic response. The parameters for the APS grid are also shown in Table VII.

The simulation was initially run using only the flow solver to get an initial flow field for the aeroelastic solver. The flow conditions were: P^*_{inlet} = 38,775 Pa,

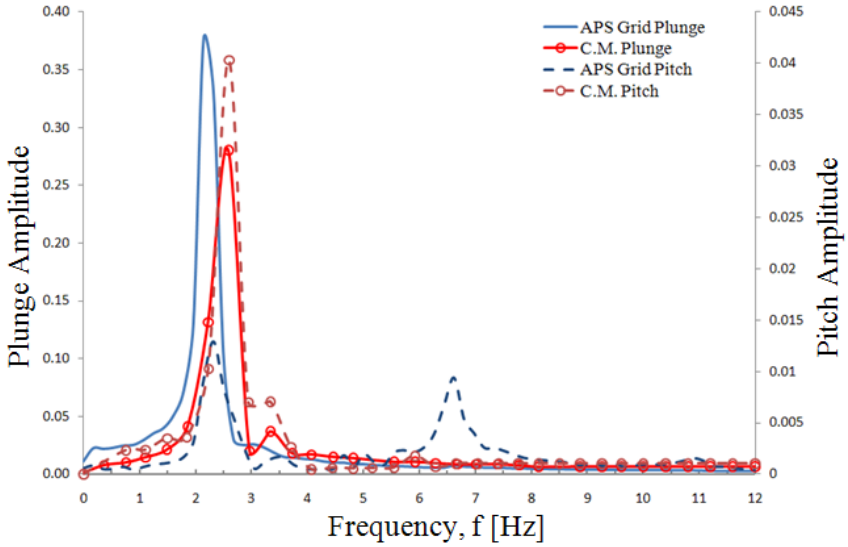


Fig. 31. Frequency for the first two modes for the conformal mapping and the APS grids.

$P_{exit} = 30{,}400$ Pa, $T^*_{inlet} = 288$ K, and $R_{gas} = 85.773$ N m/kg. These initial conditions were used to match the Reynolds number of 15x10^6 and the dynamic pressure, Q_f, of 6225 Pa (130 psf). The resultant inlet Mach number was 0.6.

Table VII. Generic Business Jet Wing grid parameters.

Grid Type	Size	I_{max}	J_{max}	K_{max}	Total Nodes	C_L	C_D
Conformal	Coarse	128	7	14	55,656	0.052	0.0298
Conformal	Medium	256	11	17	75,302	0.053	0.0184
Conformal	Fine	512	22	29	296,833	0.054	0.0193
APS	Medium	256	11	17	91,304	0.053	0.0168

The Generic Business Jet Wing was modeled as a non-linear, cantilever beam with a constant cross section. To find the properties of the beam model, the first two frequencies given in [1] were matched. The beam was assumed to be made out

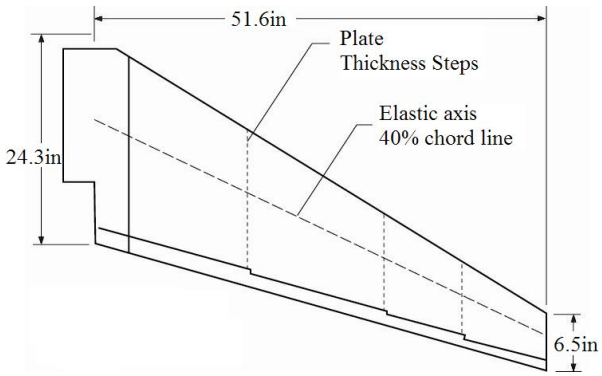
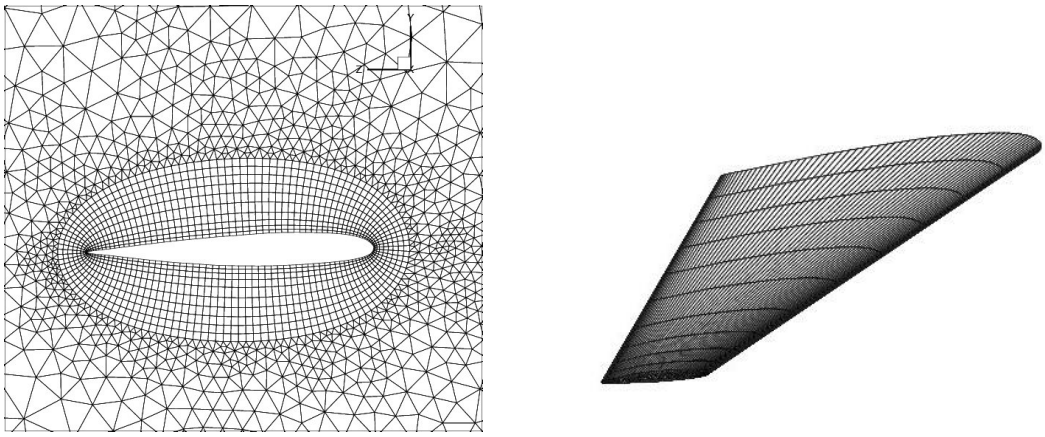


Fig. 32. The Generic Business Jet Wing planform tested in TDT [1].

of aluminum and to have a cross sectional width ten times the height. With this assumption the frequency was calculated by: $\omega = \beta^2\sqrt{\frac{EI}{\rho A}}$, where A was the cross sectional area, I was the second area moment of inertia, E was the Young's modulus, ω was the undamped natural frequency, β was related to the mode shape, and ρ was the density of the beam.

The aeroelastic simulation was run for 3.5 seconds, using both the medium conformal mapping mesh and the APS mesh. Figure 34 shows the amplitudes of the first two modes: the out-of-plane plunging mode and the pitching mode of the beam for the conformal mapping mesh. A FFT of pitching and plunging modal time coefficient history reveals the most prevalent frequency of motion as 37.5 Hz along with frequencies of 4.5 Hz, 8.1 Hz and 9.1 Hz. These frequencies are very different than the frequencies in [1] which was 10 Hz. This large error is due to the way the structure of the wing was modeled.

The pitching and plunging modal time coefficient history plots suggest that the wing was experiencing limit cycle oscillation. To verify this, the phase plots were generated and are shown in Fig. 35. As both phase plots clearly show, the Generic



(a) (b)

Fig. 33. Generic Business Jet Wing mesh (a) base layer (b) wing surface mesh.

Business Jet Wing is undergoing LCO. This was in agreement with the results presented in [1].

In addition, the simulation was attempted on a mesh built using the algebraic grid with Poisson smoothing, however, the APS grid was unable to capture the deformations without breaking the mesh. After 10 to 15 structural time steps, the mesh at the trailing edge of the O-grid was so deformed that it overlapped the O-grid. This caused negative volume cells, which stopped the aeroelastic simulation. As a result, the APS grid was not able to be used for this simulation.

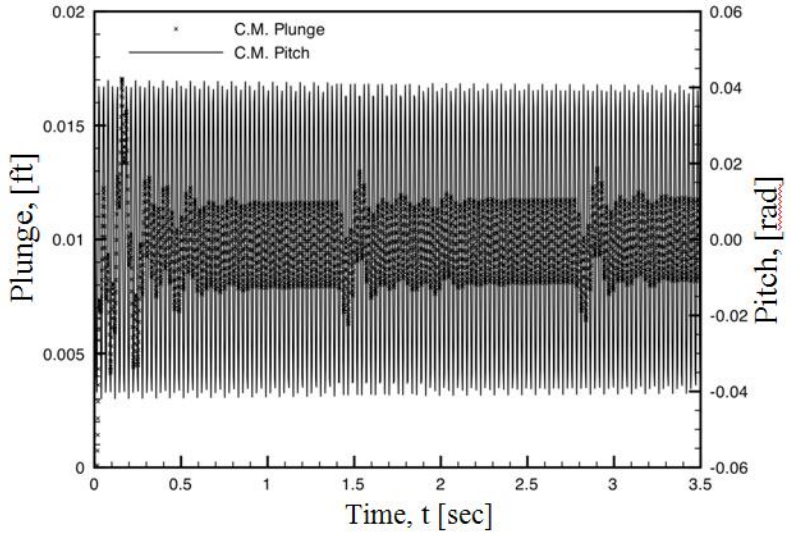
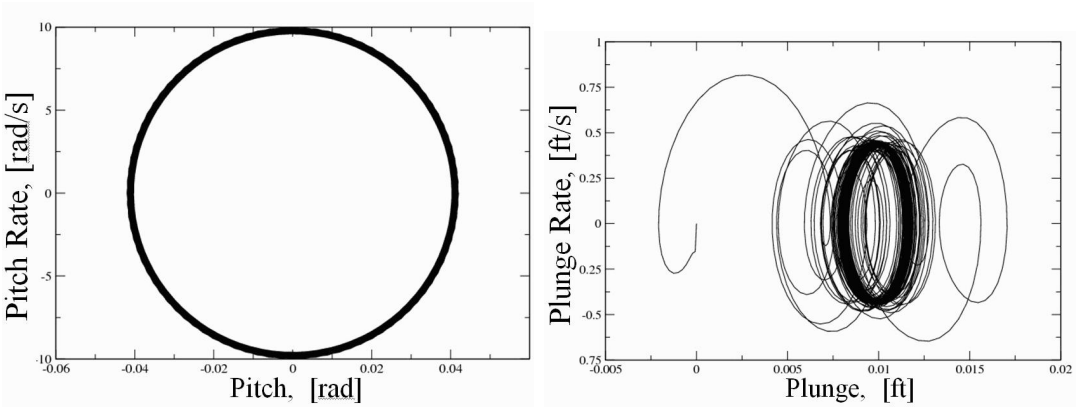


Fig. 34. Generic Business Jet Wing pitch and plunge.



(a) (b)

Fig. 35. Generic Business Jet Wing phase plots: (a) pitch and (b) plunge.

CHAPTER V

CONCLUSIONS AND FUTURE WORK

A. Conclusions

The grid generation algorithm presented in this work is an improvement upon the work of Kim [3] and Garoloff [34]. The 2D algebraic grid generation method used to generation the O-grid around the airfoil was replaced by a conformal mapping 2D mesh generation method which increased the O-grid mesh quality and robustness. By only replacing the O-grid, the advantages of the hybrid mesh is maintained and quick implementation was possible. The conformal mapping O-grid mesh generation method successfully extend the work done by Berbente and Cizmas [38] to allow for mesh generation, while retaining the method's original purpose of inviscid velocity field calculation. In addition, checks were implemented to ensure the correct velocity field calculations, and by extension, the correct O-grid mesh generation. This O-grid mesh generation method is robust and can handle complex airfoil shapes as well as deformed airfoil shapes. The 3D volume mesh generation algorithm was retained from the work by Kim [3] and Garoloff [34].

The abilities of the mesh generator were greatly expanded by the addition of a fuselage mesh generator. The fuselage mesh generator used a combination of surface mesh refinement and a modified 2.5D mapping method. The surface mesh refinement used the base layer mesh from an existing wing mesh to generate the surface mesh on the fuselage. This was done using a series of steps which included edge swapping, node addition, node deformation, and node projection. The decision to swap edges, add nodes, or deform nodes were all based on cell quality measures. This methodology produced a quality 2D surface mesh that the volume mesh could be built upon.

The volume mesh was built by projecting the rest of the wing mesh layers onto the surface mesh and deforming the nodes to allow for the fuselage. The resultant mesh is a quality mesh that can be used in both viscid or invisid flow simulations.

The conformal mapping mesh was tested in various simulations against both experimental data and numerical solutions obtained using the original mesh generation method. For each test case grid independence was achieved for the conformal mapping mesh. The NACA 0012 airfoil was tested to demonstrate the accuracy of the wing mesh in a turbulent viscous test case. While there was negligible difference between the two meshing algorithms, the experimental data were approximated well. The M6 ONERA wing was test to demonstrate the accuracy of the wing mesh in a transonic viscous test case. The conformal mapping mesh solution approximated the experimental data significantly better than the original mesh. Overall, the conformal mapping mesh captured nonlinear flow phenomena better than the APS grid for rigid wings.

The ability of the mesh to capture aeroelastic solutions correctly was also tested. Again, the conformal mapping and original mesh solutions were compared to each other. The Goland+ wing was tested to demonstrate the ability of the mesh to capture LCOs. The conformal mapping grid solution predicted the LCO at the correct frequency, while the APS grid solution predicuted flutter. A second aeroelastic test case was run using the MAVRIC wing, which is a swept, tapered wing. The wing also experiences LCO. The conformal mapping mesh predicted LCO behavior, but due to an inadequate structural solver the frequencies of the LCO were significantly off. The orignal mesh generator could not generate a mesh that could be run due to geometric limitations.

The final test case was the DLR-F6, which is a wing/fuselage configuration. Only the conformal mapping mesh was tested using this test case. This test case was run

to test the accuracy of the wing/fuselage mesh in a transonic flow regime. While the simulation correctly predicted shocks at the correct span locations, the location of the shock relative to the cord was not correct. In addition, there was a numerical oscillation that is not present in the results. These errors are the result of low fidelity at the wing tip and not enough mesh layers in the mid section of the wing.

B. Future Work

There is room for improvement to be made on the present work. An improvement to the wingtip mesh generation method is vital to correctly simulate transonic flow. This improvement can be done by extending the conformal mapping method part of the way into the wing tip. The low-fidelity at the mid span of the DLF-F6 will be addressed by running the simulation using parallel processing, which will enble the inclusion of many more layers into the mesh. Finally more research into the numerical oscillation needs to be done.

In addition, the wing mesh algorithm can be expanded to include a horizontal tail within the mesh. This mesh can then be used to add a horizontal tail to the wing/fuselage mesh. Additional improvements can be made to the wing/fuselage mesh generator. Using NURBS splines to represent the fuselage background mesh would result in a better surface mesh.

REFERENCES

[1] Edwards J, Schuster D, Spain C, Keller D, Moses R. MAVRIC flutter model transonic limit cycle oscillation test. *42th AIAA/ASME/ASCE/AHS/ASC Structures, Structural Dynamics, and Materials Conference*, Seattle, WA, 2001.

[2] Gargoloff J. A numerical method for fully nonlinear aeroelastic analysis. PhD Dissertation, Texas A&M University, College Station, TX 2007.

[3] Kim K. Three-dimensional hybrid grid generator and unstructured flow solver for compressors and turbines. PhD Dissertation, Texas A&M University, College Station, TX 2003.

[4] Dowell H, Crawley F, Curtiss C, Peters A, Scanlan H, Sisto F. *A Modern Course in Aeroelasticity*. Kluwer Academic Publishers, Dordrecht, The Netherlands, 1995.

[5] Eastep E, Olsen J. Transonic flutter analysis of a rectangular wing with conventional airfoil sections. *AIAA Journal* October 1980; **18**(10):1159–1164.

[6] Denegri M Jr. Limit cycle oscillation flight test results of a fighter with external stores. *Journal of Aircraft* September-October 2000; **37**(5):761–769.

[7] Bunton W, Denegri JM. Limit cycle oscillation characteristics of fighter aircraft. *Journal of Aircraft* 2000; **37**(5):916–918.

[8] Strganac T, Cizmas P, Nichkawde C. Aeroelastic analysis for future air vehicle concepts using a fully nonlinear methodology. *AIAA/ASME/ASCE/AHS/ASC Structures, Structural Dynamics & Materials Conference*, Austin, TX, 2008; 4575–4608.

[9] Allen C. Parallel universal approach to mesh motion and application to rotors in forward flight. *International Journal of Numerical Methods in Engineering* August 2006; **69**(10):2126–2149.

[10] Chen P, Zhang Z, Sengupta A, Lui D. Overset Euler/boundary layer solver with panel-based aerodynamic modeling for aeroelastic applications. *50th AIAA/ASME/ASCE/AHS/ASC Structures, Structural Dynamics, and Materials Conference*, Palm Springs, CA, 2009; 1–27.

[11] Chan W, Gomez RI, Rogers S, Buning P. Best practices in overset grid generation. *32nd AIAA Fluid Dynamics Conference and Exhibit*, St. Louis, MO, 2002; 1–23.

[12] Collard J, Cizmas P. The effects of vibrating blades on turbomachinery rotor-stator interaction. *International Journal of Turbo and Jet-Engines* January 2003; **20**(1):17–39.

[13] Rodriguez B, Benout C, Gardarein P. Unsteady computation of the flowfield around a helicopter rotor with model support. *43rd AIAA Aerospace Science Meeting*, Reno, NV, 2005; 1–18.

[14] Batina J. Unsteady Euler airfoil solutions using unstructured dynamic meshes. *AIAA Journal* August 1990; **28**(8):1381–1388.

[15] Khawaja A, Kallinderis Y. Hybrid grid generation for turbomachinery and aerospace applications. *International Journal for Numerical Methods in Engineering* 2000; **49**:145–166.

[16] Pizadeh S. Three-dimensional unstructured viscous grids by the advancing-layers method. *AIAA Journal* 1996; **24**(1):43–49.

[17] Owen S. A survey of unstructured mesh generation technology. *7th International Meshing Roundtable*, Dearborn, MI, 1998; 239–267.

[18] Mingwu L, Benzley S, Sjaardema G, Tautges T. A multiple source and target sweeping method for generating all-hexahedral finite element meshes. *5th International Meshing RoundTable*, Pittsburgh, PA, 1996; 217–225.

[19] Thompson J. Grid generation techniques in computation fluid dynamics. *21st Aerospace Sciences Meeting*, Reno, NV, 1984; 1505–1523.

[20] Thompson J. A reflection on grid generation in the 90s: Trends, needs, and influences. *5th International Conference on Grid Generation in CFS*, Mississippi State University, Starkville, MS, 1996; 1029–1110.

[21] Shewchuk J. Triangle: Engineering a 2D quality mesh generator and delaunay triangulator. *Workshop on Applied Computational Geometry, Toward Geometric Engineering*, Philadelphia, PA, 1996; 203–222.

[22] Rypl D, Bittnar Z. Direct triangulation of 3D surfaces using the advancing front technique. In *Numerical Methods in Engineering '96*. J. Wiley and Sons, Ltd: Hoboken, NJ, 1996; 86–99.

[23] Ghia U. An essential tool for computational fluid dynamics: Numerical grid-generation - an overview. *30th AIAA Fluid Dynamics Conference*, Norfolk, VA, 1999; 1–13.

[24] Nakahashi K, Sharov D. Direct surface triangulation using the advancing front method. *AIAA Computational Fluid Dynamics Conference*, San Deigo, CA, 1995; 442–451.

[25] Lohner R. Regridding surface triangulations. *Journal of Computational Physics* June 1996; **126**(1):1–10.

[26] Perair J, Perior J, Morgan K. Adaptive remeshing for compressible flow computations. *Journal of Computational Physics* 1987; **72**:449–466.

[27] Samareh-Abolhassani J. Triangulation of NURBS surfaces. *Technical Report*, NASA Langley Technical Report 1994.

[28] George P, Seveno E. The advancing front mesh generation method revisited. *International Journal for Numerical Methods in Engineering* 1994; **37**(21):3605–3619.

[29] Bechet E, Cuilliere J, Trochu F. Generation of a finite element mesh from stereolithography (STL) files. *Computer Aided Design* 2002; **34**:1–17.

[30] Ito Y, Nakahasi K. Direct surface triangulation using sterolithography (STL) data. *AIAA Journal* 2000; **40**(3):490–496.

[31] Lo S. A new mesh generation scheme for arbitrary planar domains. *International Journal for Numerical Methods in Engineering* 1985; **21**:1403–1426.

[32] Han Z, Cizmas P. A CFD method for axial thrust load prediction of centrifugal compressors. *International Journal of Turbo & Jet-Engines* 2003; **20**(1):1–16.

[33] Gargoloff J, Cizmas P, Strganac T, Beran P. Parallel algorithm for fully nonlinear aeroelastic analysis. *47th AIAA/ASME/ASCE/AHS/ASC Structures, Structural Dynamics, and Materials Conference*, Newport, RI, 2006; 1–11.

[34] Cizmas P, Gragoloff J. Mesh generation and deformation algorithm for aeroelasticity simulations. *Journal of Aircraft* 2008; **45**(3):1062–1066.

[35] Kim K, Cizmas P. Three-dimensional hybrid mesh generation for turbomachinery airfoils. *AIAA Journal of Propulsion and Power* 2002; **18**(3):536–543.

[36] Ivanov V, Trubetskov M. *Handbook of Conformal Mapping with Computer Aided Visualization*. CRC Press: Boca Raton, FL, 1995.

[37] Hildebrand F. *Advanced Calculus for Applications*. Prentice-Hall International, Inc.: Upper Saddle River, NJ, 1962.

[38] Berbente C, Cizmas P. Conformal mapping extension of cascade plates to cascade airfoils of a given shape. *Journal of the Romanian Academy, Studii si Cercetari de Mecanica Aplicata* 1990; **49**(5-6):466–476.

[39] Weisstein E. Quadratic Curve. mathworld.wolfram.com/QuadraticCurve.html. From MathWorld - A Wolfram Web Resource, accessed in November, 2008.

[40] Karamcheti K. Two-dimensional motion and the complex variable. In *Principles of Ideal-Fluid Aerodynamics*. Krieger Publish Co: Melbourne, FL, 1980; 456–480.

[41] Katz J, Plotkin A. *Low Speed Aerodynamics*. 2nd edn., Cambridge University Press: Cambridge, UK, 2001.

[42] Burns M. The StL format. www.ennex.com/ fabbers/Stl.asp 1989. From Automated Fabrication, accessed on February, 2009.

[43] Brodersen O, Eisfeld B, Raddatz J, Frohnapfel P. DLR results from the 3rd AIAA computational fluid dynamics drag prediction workshop. *Journal of Aircraft* May-June 2008; **45**(3):823–836.

[44] Ito Y, Nakahasi K. Surface triangulation for polygonal models based on CAD data. *International Journal for Numerical Methods in Fluids* 2001; **39**:75–96.

[45] Martin A. ADMesh version 0.95. www.varlog.com/index.html. Accessed in February, 2009.

[46] Karman S. Unstructured viscous layer insertion using linear-elastic smoothing. *44th AIAA Aerospace Sciences Meeting and Exhibit*, Reno, NV, 2006.

[47] TecPlot. Tecplot 360 user's manual. www.tecplot.com 2009.

[48] Harris C. NACA 0012 force and pressure data. *Technical Report*, Transonic Pressure Tunnel 1980.

[49] Bishop D, Noack R. An implicit flow solver with upwind differencing for three-dimensional hybrid grids. *12th AIAA Computational Fluid Dynamics Conference*, San Diego, CA, 1995; 697–710.

[50] Vassberg J, Tinoco E, Mani M, Brodersen O, Eisfeld B, Wahls R, Morrison J, Zickuhr T, Laflin K, Mavriplis D. Abridged summary of third AIAA computational fluid dynamics drag prediction workshop. *Journal of Aircraft* May-June 2008; **45**(3):781–798.

[51] Parker G, Maple R, Beran P. The role of viscosity in store-induced limit-cycle oscillation. *46th AIAA/ASME/ASCE/AHS/ASC Structures, Structural Dynamics & Materials Conference*, Austin, TX, 2005; 1–17.

[52] Beran P, Strganac T, Kim K, Nichkawde C. Store induced limit cycle oscillations using a model with full system nonlinearities. *Nonlinear Dynamics* September 2004; **37**(4):323–339.

VITA

Greg Dorway Worley received his Bachelor of Science degree in aerospace engineering from Texas A&M University in May of 2007. He entered graduate school in the Dwight Look College of Engineering at Texas A&M University in June of 2007. His research interests include nonlinear aero-elasticity, CFD analysis and mesh generation. Mr. Worley may be reached via e-mail at gdworley@tamu.edu or mailing address at 5731 Rocky Brook, Houston Tx 77345.